# REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggesstions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA, 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any oenalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

| | | |
|---|---|---|
| 1. REPORT DATE (DD-MM-YYYY)<br>01-11-2013 | 2. REPORT TYPE<br>Final Report | 3. DATES COVERED (From - To)<br>1-May-2013 - 31-Oct-2013 |

| | |
|---|---|
| 4. TITLE AND SUBTITLE<br>Effective Cyber Situation Awareness (CSA)Assessment and Training Final Report | 5a. CONTRACT NUMBER |
| | 5b. GRANT NUMBER<br>W911NF-13-C-0060 |
| | 5c. PROGRAM ELEMENT NUMBER<br>606055 |
| 6. AUTHORS<br>Steven Shope, Ph.D. | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES

Sandia Research Corporation
7565 E Eagle Crest Drive
Suite 101
Mesa, AZ 85207 -1041

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)

U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
63617-CS-ST1.1

12. DISTRIBUTION AVAILIBILITY STATEMENT

Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

The recent increase in cyber attacks against United States critical assets has greatly expanded the need for effective cyber defenses. Human cyber analysts are an essential element in these efforts. Information overload and a concomitant lack of comprehensive cyber situation awareness are common problems that hamper the effectiveness of analysis. Systems that can carry out human-in-the-loop simulation of the cyber analysis task will lead to new capabilities in
assessing the effectiveness of analysts and the support tools they use and will help enhance individual and team

15. SUBJECT TERMS

Final Technical Report

| | | | | | |
|---|---|---|---|---|---|
| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 15. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON<br>Steven Shope |
| a. REPORT<br>UU | b. ABSTRACT<br>UU | c. THIS PAGE<br>UU | UU | | 19b. TELEPHONE NUMBER<br>480-988-1000 |

## Report Title

Effective Cyber Situation Awareness (CSA)Assessment and Training Final Report

## ABSTRACT

The recent increase in cyber attacks against United States critical assets has greatly expanded the need for effective cyber defenses. Human cyber analysts are an essential element in these efforts. Information overload and a concomitant lack of comprehensive cyber situation awareness are common problems that hamper the effectiveness of analysis. Systems that can carry out human-in-the-loop simulation of the cyber analysis task will lead to new capabilities in
assessing the effectiveness of analysts and the support tools they use and will help enhance individual and team performance. This Phase I STTR effort showed the feasibility of a new capability for assessing cyber team effectiveness, cyber support tools, cyber training regimes, and the integration of multiple-component systems with human operators. We developed a novel test-bed that provides a simulation environment for the cyber analysis task and that is
equipped with measures of individual, team, and system effectiveness that allows for the assessment of cyber support tools and visualizations, cyber training regimes, and cyber concepts of operation. The effectiveness metrics embedded within the test-bed provide real and meaningful measurement of analyst performance, will aid in selecting support tools, and can be used to optimize the use of human capital through. Additionally, the test-bed can be used to evaluate and improve training protocols.

## Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

### (a) Papers published in peer-reviewed journals (N/A for none)

Received Paper

**TOTAL:**

**Number of Papers published in peer-reviewed journals:**

### (b) Papers published in non-peer-reviewed journals (N/A for none)

Received Paper

**TOTAL:**

**Number of Papers published in non peer-reviewed journals:**

### (c) Presentations

**Number of Presentations:** 0.00

## Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received Paper

**TOTAL:**

**Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):**

## Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received Paper

**TOTAL:**

**Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):**

## (d) Manuscripts

Received Paper

**TOTAL:**

**Number of Manuscripts:**

## Books

Received Paper

**TOTAL:**

## Patents Submitted

## Patents Awarded

## Awards

## Graduate Students

| NAME | PERCENT_SUPPORTED | Discipline |
|---|---|---|
| Michael Champion | 1.00 | |
| **FTE Equivalent:** | **1.00** | |
| **Total Number:** | **1** | |

## Names of Post Doctorates

| NAME | PERCENT_SUPPORTED |
|---|---|
| **FTE Equivalent:** | |
| **Total Number:** | |

## Names of Faculty Supported

| NAME | PERCENT_SUPPORTED | National Academy Member |
|---|---|---|
| Massimiliano Albanese | 1.00 | |
| Nancy J Cooke | 1.00 | Yes |
| Sushil Jajodia | 1.00 | |
| **FTE Equivalent:** | **3.00** | |
| **Total Number:** | **3** | |

## Names of Under Graduate students supported

| NAME | PERCENT_SUPPORTED |
|---|---|
| **FTE Equivalent:** | |
| **Total Number:** | |

## Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period: ...... 0.00

The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:...... 0.00

The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:...... 0.00

Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):...... 0.00

Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:...... 0.00

The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense ...... 0.00

The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:...... 0.00

## Names of Personnel receiving masters degrees

NAME

**Total Number:**

## Names of personnel receiving PHDs

NAME

**Total Number:**

## Names of other research staff

| NAME | PERCENT_SUPPORTED |
|---|---|
| Kyle Uithoven | 0.00 |
| Stephanie Lambert | 0.00 |
| **FTE Equivalent:** | **0.00** |
| **Total Number:** | **2** |

## Sub Contractors (DD882)

1 a. George Mason University

1 b. Office of Sponsored Programs
4400 University Drive, MS 4C6
Fairfax VA 220304422

**Sub Contractor Numbers (c):** 2013-OSDCyber-GMU

**Patent Clause Number (d-1):**

**Patent Date (d-2):**

**Work Description (e):** Analyze Cognitive Requirements, Identify and integrate Cyber Analysis Tools, Design sc

**Sub Contract Award Date (f-1):** 6/11/13 12:00AM

**Sub Contract Est Completion Date(f-2):** 10/31/13 12:00AM

---

1 a. George Mason University

1 b. Office of Sponsored Programs
4400 University Drive, MSN 4C6
Fairfax VA 220304422

**Sub Contractor Numbers (c):** 2013-OSDCyber-GMU

**Patent Clause Number (d-1):**

**Patent Date (d-2):**

**Work Description (e):** Analyze Cognitive Requirements, Identify and integrate Cyber Analysis Tools, Design sc

**Sub Contract Award Date (f-1):** 6/11/13 12:00AM

**Sub Contract Est Completion Date(f-2):** 10/31/13 12:00AM

---

1 a. Arizona State University

1 b. ORSPA
PO Box 876011
Tempe AZ 852876011

**Sub Contractor Numbers (c):** 2013-OSDCyber-ASU

**Patent Clause Number (d-1):**

**Patent Date (d-2):**

**Work Description (e):** Analyze Cognitive Requirements, Cyber Dextar Synthetic Task Environment, Identify an

**Sub Contract Award Date (f-1):** 5/16/13 12:00AM

**Sub Contract Est Completion Date(f-2):** 10/31/13 12:00AM

---

1 a. Arizona State University

1 b. Office of Research and Sponsored P
P.O. Box 873503
Tempe AZ 852873503

**Sub Contractor Numbers (c):** 2013-OSDCyber-ASU

**Patent Clause Number (d-1):**

**Patent Date (d-2):**

**Work Description (e):** Analyze Cognitive Requirements, Cyber Dextar Synthetic Task Environment, Identify an

**Sub Contract Award Date (f-1):** 5/16/13 12:00AM

**Sub Contract Est Completion Date(f-2):** 10/31/13 12:00AM

---

## Inventions (DD882)

## Scientific Progress

Our Phase I STTR team is very pleased with the outcome of this project. The feasilbility of a cyber warefare test-bed using validate metrics has been firmly established. In summary these are a few highlights of the Phase I STTR effort:

o Conducted a cognitive needs assessment for the cyber analyst domain
o Designed an effective cyber test-bed
o Constructed the DEXTAR test-bed using a combination of powerful servers and a large array of virtual entities
o Developed a threat injection capability for injecting a wide range of benign and hostile threats
o Custom developed and array of fully functioning metrics and measures
  ?? real-time
  ?? post-processing
o Designed an initial scenario
o Implemented this scenario
o Successfully pilot tested this scenario with humans-in-the-loop
o Successfully demonstrated this scenario in October 2013 at the MURI Cyber Workshop
o Learned valuable lessons to be applied in larger scale operations in a possible Phase II effort.

See attachment for further information and images.

## Technology Transfer

**STTR Phase I**
**FINAL REPORT**

# *Effective Cyber Situation Awareness (CSA) Assessment and Training*

**Submission Date: 1 November 2013**
**Contract Number:  W911NF-13-C-0060**
**Period of Performance:  1 May 2013 to 31 October 2013**
**Solicitation #:  DOD FY2012.B**
**Topic #:  OSD-T08**
**Phase I Proposal #: 012-B-T08-2022**

By:

**Steven M. Shope, Ph.D.**
**Principal Investigator**
**Sandia Research Corporation**
**7565 Eagle Crest Drive**
**Mesa, AZ 85207**
**(480)988-1000**

# NOTICE AND SIGNATURE PAGE

This report is published in the interest of scientific and technical information exchange and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

**Steven M. Shope, Ph.D. Principal Investigator**

# ABSTRACT

The recent increase in cyber attacks against United States critical assets has greatly expanded the need for effective cyber defenses. Human cyber analysts are an essential element in these efforts. Information overload and a concomitant lack of comprehensive cyber situation awareness are common problems that hamper the effectiveness of analysis. Systems that can carry out human-in-the-loop simulation of the cyber analysis task will lead to new capabilities in assessing the effectiveness of analysts and the support tools they use and will help enhance individual and team performance. This Phase I STTR effort showed the feasibility of a new capability for assessing cyber team effectiveness, cyber support tools, cyber training regimes, and the integration of multiple-component systems with human operators. We developed a novel test-bed that provides a simulation environment for the cyber analysis task and that is equipped with measures of individual, team, and system effectiveness that allows for the assessment of cyber support tools and visualizations, cyber training regimes, and cyber concepts of operation. The effectiveness metrics embedded within the test-bed provide real and meaningful measurement of analyst performance, will aid in selecting support tools, and can be used to optimize the use of human capital through. Additionally, the test-bed can be used to evaluate and improve training protocols.

# TABLE OF CONTENTS

**Section** **Page**

# LIST OF FIGURES

# LIST OF TABLES

# PREFACE

This project was sponsored by the U.S. Army Research Office (ARO). Dr. Cliff Wang ARO was the Program Manager for this effort. The contract was awarded in response to a STTR proposal submitted under the 2012 FY2012.B solicitation, Topic OSD-T08: *"Effective Cyber Situation Awareness (CSA) Assessment and Training."*Sandia Research Corporation was the Prime Contractor for the effort with Institution partners Arizona State University and George Mason University. We acknowledge the valuable discussions concerning this project that we had with Dr. Wang.

# 1. INTRODUCTION

## 1.1 The Problem

The rapid growth of cyber threats coupled with the information overload associated with the cyber analyst task has resulted in a wide variety of commercial off-the-shelf (COTS) hardware and software tools and components for use in cyber defense by the Department of Defense (DoD). The downside of using COTS components is that the resulting system of separate components is often more complex, harder to make secure, and based on information technology that is equally available for adversaries. More importantly, the component tools may not be well integrated with the needs of the cyber analyst and may go unused or worse, interfere with the analyst task.

The recent increase in cyber attacks against United States critical assets has greatly expanded efforts to develop effective cyber defenses. The term cyber situation awareness has been used to refer to the analyst's ability to comprehend the status of cyber assets critical for achieving mission success. A critical requirement for cyber situational awareness is to understand the overall context of network vulnerabilities, how they are interrelated, and how attackers may exploit them to penetrate deeper in the network.

Human cyber analysts are an essential element in these efforts. Information overload and a concomitant lack of comprehensive cyber situation awareness are common problems that hamper the effectiveness of analysis. There are many parallels to the cyber defense problem with other military tasks such as real time imagery analysis. One way to improve a cyber defensive system is to enhance the effectiveness of cyber analysis and to increase their performance, especially at the team level. This requires a clear understanding of cyber situational awareness, an ability to measure it, and a capability for evaluating the extent to which tools, visualizations, and proposed strategies impact cyber situation awareness.

Further, the rapidly growing and important cyber defense initiative and the growth of network-centric operations have created a need for highly-responsive, networked teams of individuals and autonomous systems to collaborate in order to solve time-critical cyber defense problems. The effective evaluation of these complex systems requires system-level, operator-level, and team-level metrics. Unfortunately, these system, operator, and team metrics largely do not exist.

Support technologies such as attack graph technology and operational concepts like interdependent team structures can help create a common operating picture and lay a foundation for human analysts to improve cyber situation awareness. However, the rapid proliferation of these tools in the commercial market place makes tool evaluation essential. In particular, questions of how well does a tool improve a human-machine cyber defense is a critical question. Some tools can actually degrade performance.

Systems that can carry out human-in-the-loop simulation of the cyber analysis task will lead to new capabilities in assessing the effectiveness of analysts and the support tools they use and will help enhance individual and team performance. Simulation environments provide ideal solutions to lack of ground truth found in the real world which hampers our ability to assess human performance. These simulations can also assist in evaluating concepts of operation in the analysis process or training regimes for new analysts. Further human-in-the-loop simulation can be used to test how a set of analyst tools can be integrated and work together in service of the human analyst (i.e., good human systems integration).

The effort proposed in this Phase I effort will create a new capability for assessing cyber team effectiveness, cyber support tools, cyber training regimes, and the integration of multiple-component systems with human operators. The effectiveness metrics embedded within the simulation system can provide real and meaningful measurement of analysts performance (given ground truth), can aid in selecting support tools, will optimize the use of human capital (the  most costly part of cyber defense), and will provide a better training protocol for training and enhancing the performance of less experienced analysts.

## 1.2 Cyber Analysts

Cyber analysts are an essential component of any effective cyber defensive system. Analysts must determine if cyber activities taking place are threatening given the context of the situation. The ability to recognize and relay information regarding potential threats is vital for successful network operations. Currently, training focuses on known isolated threats with little to no emphasis on contextual interpretation.

Cyber analysis is a difficult cognitive problem and there are issues that make cyber threat detection even more challenging. Because cyber threats can come in many forms and can be hidden virtually any type of network traffic, direct detection of a threat may be difficult to impossible. The range of cyber threats is vast and attacks are constant. Threats can be dynamically changing. Threats may be immediate or they may unfold over time and may consist of multiple and diverse threats attacking synchronously. Lethal threats must be detected in often high levels of non-lethal threats that are ubiquitous on most networks. This is a classic signal in noise detection problem. Additionally, these threats are often from an adversary that is constantly trying to thwart detection and thus, employing tactics in which the threat signature is constantly changing. Cyber threat perception is often not direct, but is filtered through cyber tools and graphical displays. However, despite the immense challenges inherent in this task, human cyber analysts do detect cyber threats, they acquire skill at the task, and they can train others in threat detection. Cyber-support tools will not replace the analyst in the foreseeable future.

## 1.3 Project Overview

This Phase I effort was directed at developing a novel test-bed that provides a human-in-the-loop simulation environment for the cyber analysis task and that is equipped with measures of individual, team, and system effectiveness to allow for the assessment of cyber support tools and visualizations, cyber training regimes, and cyber analysis concepts of operation.

Our Phase I team consisted of Dr. Steven Shope from Sandia Research Corporation, Dr. Sushil Jajodia from George Mason University, and Dr. Nancy J. Cooke from Arizona State University was ideally and uniquely positioned to carry out this project. Dr. Shope and Sandia Research Corporation bring fifteen years of test-bed design and development to the table along with capability to implement custom metrics of individual, team, and system performance. Dr. Sushil Jajodia brings Cauldron, a state-of-the-art tool for cyber analyst support that will be integrated and evaluated in the context of the test-bed. Dr. Cooke and her team at Arizona State University have done over fifteen years of experimental research on teams and systems in the context of synthetic task environments. The ASU team has recently developed CyberCog, a test-bed that simulates the analyst triage task of discriminating true from false alerts and is collecting data in that  test-bed currently. Dr. Cooke has also conducted cognitive task analyses at several cyber exercises and through an on-line survey and specializes in the development of metrics, for individuals, teams, and system performance and process. Together, this team worked well together to provide all of the necessary components which resulted in delivering a useful cyber test-bed for cyber warfare research and assessment of tools and training to DoD.

The resulting cyber test-bed is called DEXTER. This is an acronym for **D**efense **EX**ercises for **T**eam **A**wareness **R**esearch.

## 1.4 Phase I Objectives

The objectives of our Phase I effort are described here. Note that these tasks were highly iterative.

**1.4.1 Objective A – Analyze Cognitive Requirements** Identify the cognitive requirements of the cyber analysis scenarios that will be represented in the test-bed. Integrate with the requirements of cyber analysis tools (i.e., Cauldron) through use case scenarios.

**1.4.2 Objective B – Design and Develop Synthetic Task Environment** Design a synthetic task environment using DEXTAR and based on the findings of Objective A.

**1.4.3 Objective C – Identify and Integrate Cyber Analysis Tools** How can tools such as network models, graphics, vulnerability information and decision support aids be effectively used to augment the functions of a human-in-the-loop defensive cyber analyst?

**1.4.4 Objective D – Identify and Integrate Metrics** Extend individual and team metrics for use in a cyber domain.

**1.4.5 Objective E – Design Scenario and Collect Preliminary Validation Data** Design an initial set of STE scenarios.

# 2. COGNITIVE REQUIREMENTS ANALYSIS

Because the DEXTAR system with embedded scenarios is intended to exercise cognitive processing on the part of participants that parallels the cognitive processing required of cyber defense operators, it was necessary to identify the cognitive requirements. At the highest level it is critical that DEXTAR be capable of simulating:

- Advanced cyber defense (i.e., beyond triage)
- Vertical as well as horizontal information sharing
- Use of tools such as Cauldron

In addition to these high-level tasks, DEXTAR needs to be capable of exercising the cognitive processing involved in these tasks. To identify the cognitive requirements of DEXTAR we leveraged data from our existing cognitive analyses (observations of cyber exercises, interviews with expert operators, and survey of operators) and explored more advanced cyber analysis tasks by interviewing cyber operators to lay out the cognitive requirements of the cyber scenarios.

## 2.1 Existing Cognitive Analyses

Earlier observations of a Capture the Flag competition at University of California, Santa Barbara (UCSB) demonstrated the important role of collaboration and communication in successful cyber operations (Jariwala, Champion, Rajivan, & Cooke, 2012). In addition, observations from this exercise inspired the transition from our CyberCog test-bed to DEXTAR.

Our on-line survey data also provided information useful for developing cognitive requirements. In part of our survey we asked 119 respondents how many hours a day they spend looking at network traffic alerts. Figure 1 shows the time spent on cyber analysis related activities. The highest peak is 3-4 hours with 24 respondents. When looking at the information security officers and managers group alone, the number of hours a day spent on cyber analysis related activities" averages 4.78 hours with a similar peak at 3-4 hours.

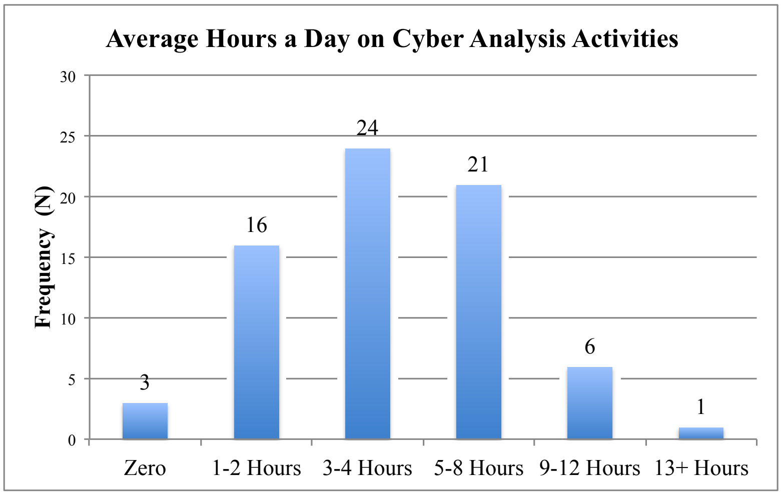


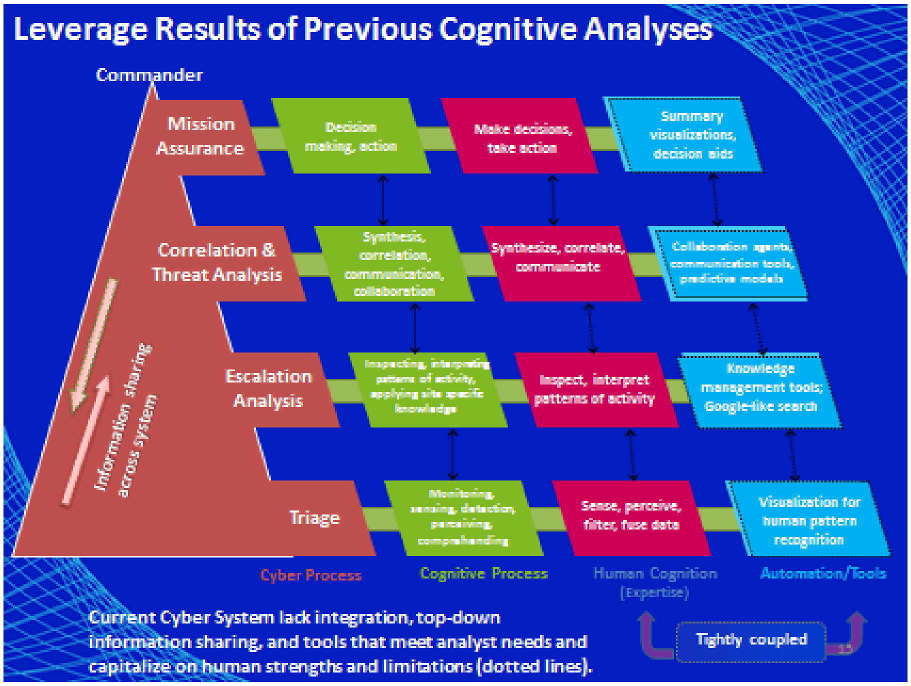
**Figure 1. Average hours a day on cyber analysis activities. Most people spent only a few hours a day on activities related to cyber analysis.**

In this same survey, many of the respondents indicated that they work using multiple operating systems. Out of the 119 responses, 55 used a Windows End User system, 47 used a Linux End User system, 34 used a Linux Server, 27 used Mac OS X, 26 used Windows Server and 26 used Unix-based Systems. There were some respondents who did their cyber work using Unix-based servers (14), OS X servers (4) and other systems (8). Other systems included IBM Mainframes and the mobile platforms Android and iOS.

Respondents were also asked about the tools and aids they use in order to complete their cyber work. The top aids to this work were search engines such as Google, newsfeeds/websites, cyber-related news feeds, cyber security forums, and blogs. Google was reported on average to be used at least a few times a day whereas the other top tools were used at least a few times a week on average by respondents. Two respondents mentioned twitter as an important aid in “other”.

Finally, when asked what impeded their work the most, analysts strongly indicated organizational issues with 55 of them ranking it within their top three issues. The lack of knowledge (43) and information overload (41) were second and third most ranked. The “Other” write-in category included a “lack of a big picture” and “lack of funds and personnel”. A more even picture was drawn when asked for the top potential improvements: organizational change was at top (46), followed by more training (43), better tools (41), better collaboration (37), more information (29) and other (10) which was often more funding and personnel.

As a result of our earlier cognitive task analyses we developed an understanding of cyber security operations as a cognitive system (Figure 2). The systems can also be thought of as a sociotechnical system that involves multiple operators working with multiple types of technology. Situation awareness in this context involves the entire system. Whereas our early test-bed, CyberCog, focused purely on triage, DEXTAR will also be including the other systems functions of escalation analysis, correlation and threat analysis, and mission assurance.



Leverage Results of Previous Cognitive Analyses

Commander
Mission Assurance
Decision making, action
Make decisions, take action
Summary visualizations, decision aids
Correlation & Threat Analysis
Synthesis, correlation, communication, collaboration
Synthesize, correlate, communicate
Collaboration agents, communication tools, predictive models
Escalation Analysis
Inspecting, interpreting patterns of activity, applying site specific knowledge
Inspect, interpret patterns of activity
Knowledge management tools; Google-like search
Triage
Monitoring, sensing, detection, perceiving, comprehending
Sense, perceive, filter, fuse data
Visualization for human pattern recognition
Information sharing across system
Cyber Process
Cognitive Process
Human Cognition (Expertise)
Automation/Tools
Tightly coupled

Current Cyber System lack integration, top-down information sharing, and tools that meet analyst needs and capitalize on human strengths and limitations (dotted lines).

**Figure 2. Cyber security as a complex sociotechnical system.**

The DEXTAR scenario being currently designed for the test-bed involves a majority of the facets of the following known cyber security task aspects:

1. Triage – the sifting through network logs, sensor logs, and security logs looking for intrusions.
2. Escalation Analysis – Answers where the attack is likely going, and what is currently happening
3. Correlation and Threat Analysis – This is identifying how much of a threat is imposed by the attack
4. Mission Assurance – maintaining system operations to ensure a mission goal is upheld.

The DEXTAR scenarios will have a missing component within the Correlation and Threat Analysis in which analysts often "Google" attacks for more information. Because of the air-gapped test-bed, this information will need to be provided or non-existent leaving the analysts constrained with respect to Google searches. All other aspects of the cyber analyst task will be supported.

1. Analysts will need to sift through logs to find intrusion alerts.

2. Analysts will have to determine which alerts are true and the most important.
3. Analyst will have to identify where the attack has been/come from and where it is likely to go next.
4. Analysts will need to stop the attack and remedy the situation in order to avoid future similar attacks.
5. Analysts will have to ensure that any remediation and attack does not affect the mission.

Cauldron is specifically built to analyze network information to identify attack vectors. This will help analyst nearly exclusively within the escalation analysis and partially within the correlation and threat analysis. This latter aspect within the correlation and threat analysis will be useful in determining if a path of attack is currently on course towards a server/cluster. Cauldron will specifically aid in situation awareness and knowledge of the network within the escalation and threat analysis stages.

## 2.2 Direct Interview

The direct interview is one of the more valuable methods for collecting specific information regarding task. This process requires that an interviewer speak directly with an individual regarding the specific topic or goal of the research project. Often times, this is a subject-matter expert (SME) but under specific circumstances the interview of novices and experienced individuals are also required, or even preferred. By utilizing SMEs a well-informed body of information can be derived with a high level of accuracy and authority. Interviewing a SME often results in directed information that has been honed over the course of time, often in the magnitude of years, and can often be considered less error prone with regards to the current standards. However, by utilizing individuals below the SME level, it is possible to extract useful information, even if the authority and accuracy of the information is lowered. Information from novices and experienced individuals within a domain are often vital at understanding what is currently perceived as important within the field, common interpretations of information, as well as a snapshot into the development of knowledge and experience within selected area. Often, the differentiation between SMEs and novices can illustrate aspects of a system that should be addressed in order to increase the efficiency of said system.

In February of 2011, the Arizona State University CERTT Lab hosted an Army Research Office sponsored workshop on Cyber Situation Awareness (see Cooke, et al., 2011, for more information). This workshop gathered researchers, SMEs, and experienced analysts into what became an open forum for discussion about cyber security from the analyst point of view. A great deal of information was gathered regarding the structure of the cyber defense task, the organization of the analysts within a structured hierarchy, current needs of the analyst, and current challenges faced by both the analyst and the domain. Although the interviews within this workshop were unpublished, this started foundational work for a number of other research advancements – one of which is DEXTAR.

The direct interviews from the ASU workshop highlighted a few aspects of the cyber defensive task that were regarded as important: availability of information, collaboration (for more information see(Jariwala, Champion, Rajivan, & Cooke, 2012)), usable tools, information overload, situation awareness, and cognitive fatigue (for more information see(McNeese, Cooke, & Champion, 2011)). Given the importance placed on these areas, and the need for research materials to properly examine these aspects, the DEXTAR system was built to examine these aspects.

## 2.3 Additional Cognitive Analyses

In addition to leveraging prior cognitive analyses of the cyber domain, we examined documentation pertinent to DEXTAR and Cauldron (Cauldron documentation, journal articles, training material, software/hardware instructional information, technical reports, and trade publications) to provide relevant information on the specific tasks we have identified. This information helped to prepare interview questions for subject matter experts (see Table 1).

Structured interviews were conducted of two leaders of cyber operations in two different organizations (industry and military). They were interviewed by phone and asked the questions listed in Table 1. Results are being modeled using the EAST (Event Analysis of Systemic Teamwork) framework (Stanton, Baber, & Harris, 2012). An example of the task network derived for the military cyber defense organization is presented in Figure 3.

**Table 1. Questions for SME with Knowledge of Cyber Organization**

| | | |
|---|---|---|
| **OPERATORS (SOCIAL)** | | |
| | • How big is the team of operators?<br>• What are their roles? | |
| | | o How many individuals are in each role |
| | • Do people collaborate? | |
| | | o Who?<br>o How often? |
| | • Can you provide an organizational chart? | |
| | | o If so, how does the chart differ from a social network chart based on collaboration?<br>o How rigid is this structure?<br>o How interchangeable are people?<br>o How often does staff rotate out of positions?<br>o How are shift handoffs accomplished? |

**TASKS**

- What are the goals of the team/organization?
- What are the main tasks to achieve these goals?
- Who does what tasks?
- How often are tasks done?
- Are some tasks more important than others?
- Are the tasks interrelated (e.g., sequentially dependent, parallel)
- Are some temporally constrained (i.e., urgent)

**INFORMATION**

- What information is needed to do each task? By whom (which role)?
- Where/Who (which role) does the information come from?
- What tools are needed to do each task?

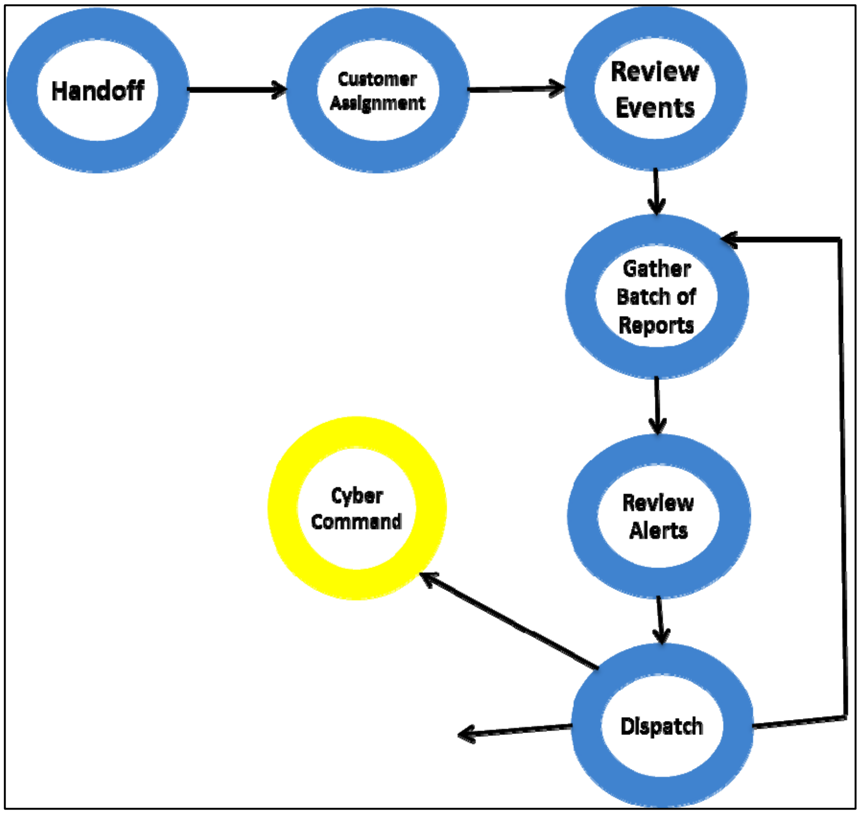


Figure 3. Task network diagram from military network defense team.

## 2.4 Cognitive Requirements

Based on the information collected in the structured interviews and previous cognitive analyses we have identified the cognitive requirements for SEXTAR in Table 2 below.

Table 2. Cognitive Requirements for DEXTAR.

| *Cognitive Requirement* | *Design Implications* |
|---|---|
| Complexity: DEXTAR needs to host task scenarios that involve multiple cyber functions: triage, escalation analysis, correlation and threat analysis, mission assurance | The task will be constructed so that situations are not easily solved to allow for situation evolution. Situations will also need to be embedded that allow for each task to be addressed. |
| Network Consequences: Structure: DEXTAR will need to provide an environment where individual tasks have consequences that can proliferate through the network | The task will be realistic such that the events that are not mitigated will lead to additional events. |
| Teaming/Collaborating: DEXTAR needs to host scenarios that require multiple analysts to interact. | DEXTAR will provide methods of sharing voice, text, and data communication (via public folder). Audio and video will be recorded as well as text chat. |
| Uncertainty: DEXTAR will have uncertainty built into the task to increase difficulty. | The Nessus scan can be edited to that it is not perfect or the scan can be done with firewalls in place. This allows for collection of data from teams working to mitigate rather than prevent attacks. |
| Specialization: DEXTAR needs to provide a means of compartmentalizing aspects of the task to simulate specialized expertise | The test-bed will need to have the ability to house several specializations. When the network gets to be > 1000 machines it will be possible to present each operator only part of the overall network for monitoring. |
| CAULDRON: DEXTAR scenarios need to be such that identification of attack vectors is a reasonable approach. | Adequate network complexity is required to best test Cauldron. |

# 3. Design of Synthetic Task Environment

Several processes influenced the design of the DEXTAR task environment. First, was the ongoing cognitive task analysis (CTA) (as described in the previous chapter) into the domain of cyber defense aided the project in understanding the basic fundamentals of the task to be emulated. Second, the development of a previous generation software platform dedicated to the study of cyber defense (Cyber-COG). Third, the identification and application of current knowledge from ongoing data collection within human-aspect cyber defense research. Lastly, to develop an environment capable of responding to current research questions with the ability to be flexible enough to be able to answer questions not yet asked.

Although the aforementioned steps can seem surprisingly disparate, each step was integral in advancing the test-bed into the current developed state. In Figure 4 we highlight the process taken step-by-step in order to achieve the current design.

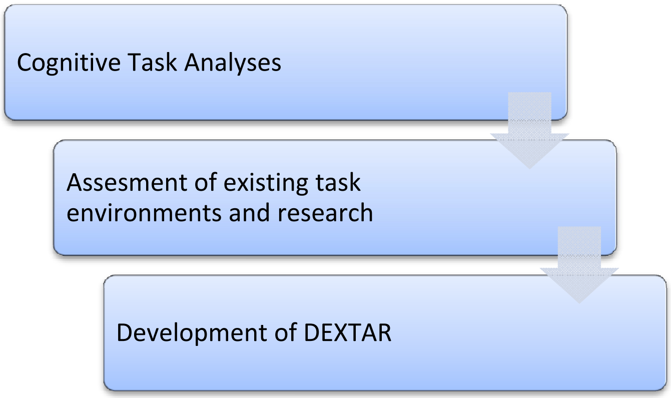


**Figure 4. Development Process of DEXTAR.**

## 3.1 Assessment of Existing Test-beds and Research

The old adage, “Don’t reinvent the wheel” was an appropriate consideration when designing the DEXTAR test-bed. Instead of striving to create an entirely new system with no previous work on this scale, existing test-beds were adapted for this system. Specifically, two systems where heavily relied upon in designing the framework for DEXTAR. These are: the CyberCog Framework (for more information, see (Rajivan et al., 2013)) and the iCTF structure as it existed in 2012 (for more information, see http://ictf.cs.ucsb.edu/). By adapting strengths from both test-beds a cohesive system was designed with aid from the aforementioned cognitive task analyses.

However, it is not enough to say that DEXTAR was designed solely on research conducted by ASU, but also through existing literature and research. Current literature was reviewed in order to better understand the wider array of tasks that our CTA may not have captured.

We utilized reports from large security agencies such as Verizon and McAfee, as well as additional CTAs and case studies within cyber security (e.g. (Byres & Lowe, 2004; D'Amico et al., 2005; Garrison & Ncube, 2011)). The information gathered from the literature search allowed us to develop potential methods in which DEXTAR could function.

## 3.2 Existing Test-beds

Before the DEXTAR project began, the ASU CERTT Lab observed the iCTF competition in Santa Barbara. Although a multitude of information was procured from this observation the idea of utilizing the architecture employed within the competition was discussed as a method for testing human performance. There were a few main aspects of the iCTF structure that were borrowed and applied to what became the DEXTAR framework, and they are:

1. A rapidly deployable virtual structure
2. A consistent template of a network structure, easily re-deployable from a generated template or copy
3. The complexity and ecological validity of a network and reliant services
4. A method for testing skill and performance
5. The multiple level approach to cyber security

The selection of these components came about from interviews of the system administrators of the competition, best practices of experimentation, and the synthetic task environment framework (Cooke & Shope, 2004).

From the above components and the literature two main concepts were derived that were the driving force into DEXTAR. The first is that DEXTAR must be a scalable virtual environment running a fully realized and functional network, and the second that the test-bed must be geared towards answering the human aspect of cyber security while being simultaneously able to investigate the computational and programming side of cyber security. These two concepts led to the design of a virtual system that is capable of being built, tested, stressed, and then reset between experimental runs with the network being 100% replicable.

After the initial concept of the framework for the DEXTAR project was developed from the iCTF competition, additional test-beds were examined for possible contribution to the project. The next test-bed that was analyzed for inclusion was the CyberCog framework (for more information, see (Champion, Rajivan, Cooke, & Jariwala, 2012; Rajivan et al., 2013; Venkatanarayanan, 2010)) . This environment was uniquely different from the iCTF framework in that the components within the test-bed were static, scripted, and lightweight. While CyberCog was primarily concerned with the entry level of triage into a large scale cyber security this did provide two components:

1. A framework for attaining a measure of situation awareness

2. Algorithms and methodologies for attaining performance measures

Within the CyberCog system participants are asked to classify alerts they receive. This classification revolves around identifying if the alert is malicious or benign. While this act alone is more akin to knowledge elicitation, a follow-up report requires participants to complete a second report that centers on their situation awareness of the alerts that occurred. The secondary component to the classification system allowed for a background algorithm to calculate performance utilizing a non-parametric signal detection method to derive a single performance score. This score was accompanied by post-hoc analytical techniques of communication and collaboration to generate an overall team score that each team was then analyzed by for conditional differences. The addition of these metrics into the DEXTAR framework provides the much-needed experimental analytics required of such an advanced test-bed.

## 3.3 Related Research

Throughout the construction of DEXTAR and before, the research within cyber security has been monitored for useful additions into the DEXTAR framework. A concern that was uncovered during the literature review for DEXTAR was the startling lack of research that attempted to replicate real-world situations for the testing of human-centric metrics. Numerous papers were found concerned the algorithmic development of cyber security, but it seemed that the human was left out of the loop. Considering this lowered level of resources to draw from, it became even more imperative to ensure that DEXTAR reaches the goal of being a human-centric testing system for cyber security. Although far more resources influenced the DEXTAR project than can be listed here, below is a summary of a few main resources that heavily influenced the design of DEXTAR.

A primary source that influenced not only DEXTAR but also the CTA groundwork was a paper by Anita D’Amico and colleagues (2005). This paper outlines the six stages within the cyber security task, and those are:

1. Triage – the base level work in cyber security were classification of intrusion detection alerts make up nearly the entirety of the task. This level is often responsible for flagging alerts that potentially highlight infiltrations, as it is the first level.
2. Escalation Analysis – a secondary level of triage analysis whereby the alerts are once again looked at in a much larger scale.
3. Correlation Analysis – this level searches for patterns within the flagged alerts.
4. Threat Analysis – this level utilizes additional resources to understand possible threats that could be occurring.
5. Incident Response Analysis – acts on, or recommends actions on any discovered incidents.
6. Forensic Analysis – gathers and preserves evidence.

The levels noted by D'Amico et al. were verified by the CTA that ASU conducted as well as being a known structure for cyber defense within military applications.  These levels are introduced into the DEXTAR system and are deployable when experimentally required.

Secondary sources involved with the design process of DEXTAR were reports on cyber security breaches and the state of the Internet reports. A yearly report released by Verizon (e.g. (Verizon, 2012; 2013; Verizon, Politie, & Service, 2011)) highlights the past years security breaches in an in-depth format. While the DEXTAR system at this phase is incapable of producing an enterprise network that would satisfy some of the case examples with the reports, several methods of entry and infiltrations highlighted throughout the reports are replicable within DEXTAR. The main component derived from these reports is the type and overall procedure and motivation of infiltration attacks. These profiles aided in the development of the example scenario that is currently installed in DEXTAR. Specific examples of breaches such as the Shady RAT breach discovered by McAfee ((Alperovitch, 2011)) were considered for components to include within DEXTAR. However, specific case studies were left to lend aid to the development of future scenarios.

## 3.4 Conclusion

In summation, DEXTAR was designed around the highlight material within this section and with these specific components in mind:

1. Produce a highly customizable virtual environment
2. Reproduce real-world network installations
3. Rapidly deployable test-bed
4. 100% Replicable network structures, machines, and scenario designs
5. Emulate the six stages of the cyber security task
6. Represent real-world situations and scenarios
7. Implement current and experimental software to be assessed for the viability of aiding the cyber security analyst
8. Collect and produce performance metric data, collaboration data, situation awareness data, and communication data.

# 4.0 Construction and Development of the STE

Given the defined parameters outlined in the previous section, the DEXTAR project built the DEXTAR Test-bed. The DEXTAR system consists of a paired set of servers running a virtualized environment that has been customized based on the literature review and conducted CTA. In the following sections, an overview of the DEXTAR system will be presented alongside a brief example of the configuration. For more information about the system including more technical details, please refer to the accompanying Manual of Operations located in the Appendix.

## 4.1 Hardware

The test-bed operates on a paired set of customized Dell R420 Servers that were procured under this project. Each server contains 2 Intel Xeon CPU chips each with 6 cores and hyper-threading architecture, effectively creating a total of 96 logical processors. Each processes runs at 1.9GHz totaling 182.4GHz in available processor speed.  The main runtime server contains 48GB of RAM, while the storage and monitor server operates with 16GB of RAM. This differentiation was customized based on resource requirements to run the DEXTAR Example Scenario. Each server also houses a CD/DVD drive, 6 USB ports, and 2 VGA ports. Lastly, each server maintained a 2TB RAID 1, which provides a level of redundancy by utilizing 2 hard drives that are identical. This provides a failsafe in the even a single drive fails.

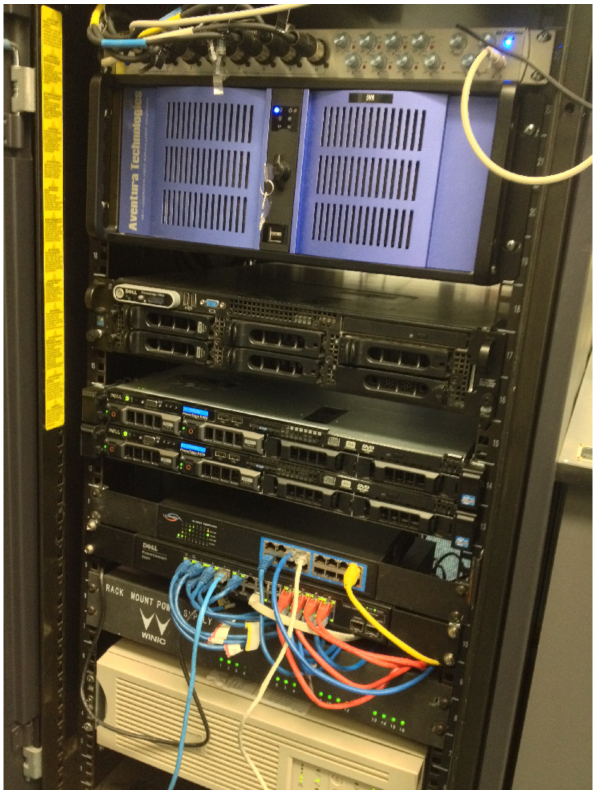


Figure 5.  DEXTAR Servers, network switch and DVR.

In order to maintain a high volume network each server is fitted with 4 network interface cards, each running at 10/100/1000 mbps. This redundancy allows for overflow traffic between each NIC card if the server receives high levels of traffic that are capable of overloading a single NIC card. This ensures no network speed loss, no network traffic loss, and complete data collection within the server between the participants and the run-time environment. To support the servers and full test-bed, a Dell PowerConnect 2824 Web-Managed Switch was installed. This switch operates at 100/1000 mbps. With these network components configured, the network latency was approximately between 500 and 1000 nanoseconds.

Although a portion of the DEXTAR test-bed was pre-existing, hardware upgrades were provided to the participant stations that interact with the DEXTAR servers. Six participant stations, two experimenter stations, and a DVR computer were used within the DEXTAR test-bed. Each of the participant and one experimenter desktop computers are a Dell Optiplex 740 with an AMD Dual Core 2.61 GHz Processor with 2GB of RAM. The remaining experimenter station is a Dell Optiplex 745 Intel Core 2 at 1.8 GHz with 2GB RAM and an 80GB Hard drive. This machine was also responsible for recording audio from a PreSonus Pre-amplifier receiving 6 noise-canceling temple mounted microphones at each of the participant stations. The DVR computer is an Aventura Custom Built Computer Intel Core 2 Duo 2.21 GHz, 1GB RAM, and a 500GB hard drive. The DVR received video input from 7 overhead cameras powered by a rack-mounted power supply. An audio feed from the Audio Pre-Amp to the DVR provided the sound attached to videos.

## 4.2 Software

The main runtime software to create the DEXTAR environment was VMware's vSphere Essentials Kit. Within this software suite, the ESXi operating system was installed onto each of the two servers, naming them Phobos and Deimos. Installed on Phobos was the main virtual management component, vCenter. This virtual appliance controlled the operation of both servers in order to interlink the networks between them. This allowed for the DEXTAR scenario to be spread across two servers, as needed. This appliance also allowed for the automation utilized within DEXTAR through command scripting.

Each of the six participant machines had Ubuntu 12.04 LTS installed onto it for the main operating system.  This OS provided a lightweight, low-overhead OS leaving additional processing power for the ability to run local virtual machines. The VMware Player was installed on each participant machine, and the main experimenter control machine, which allows for the running of a virtual machine either from one of the servers or from local hosting. TeamSpeak software was installed allowing experimenters to customize communication methods if desired by either restricting communication, or requiring communication in a certain modality. Wireshark was installed onto each machine as a method to capture all network traffic that entered and left the machine. SSH and Remote

Desktop Connection packages were also added to each machine in order to remotely operate or monitor the computer.

The DVR operated on a Windows XP installation with a customized Aventura technologies DVR recording software suite. This suite allowed for the recording all video feeds, and the integration of audio into the feed.

The audio recording computer operated on a Windows XP installation and utilized the CuBase 4 audio recording software. Special templates were built specifically for this test-bed in order to capture the audio of each participant for later analysis.

## 4.3 Virtualization

The primary method of utilizing the DEXTAR test-bed is through the virtual environment. In total, DEXTAR contains 90 virtual machines, 60 of which are dedicated to the scenario developed. Within this environment six operating systems were used in the virtual machines:

1. BackTrack 5
2. Vyatta
3. Windows XP
4. Windows Server 2008
5. Ubuntu 12.04 LTS
6. Windows 7

BackTrack 5 was utilized to generate the cyber threats used within DEXTAR. This software package is attainable from the Offensive Security website for BackTrack (http://www.backtrack-linux.org/). The Vyatta appliance is responsible for the network structure by acting as virtual routers. This appliance may be downloaded from the Vyatta community website (http://www.vyatta.org). The Windows XP was used to represent client machines in which the standard user within the network would use in our scenario. The Windows Server 2008 installations were used to act as the scenario servers and the intrusion detection system. However, these virtual machine servers only maintained control within the scenario network established by the VMware environment. The Ubuntu 12.04 LTS OS was used in both the metric and administration portion as well as the scenario network. Outside of the scenario, these OSes were responsible for generating network traffic and capturing received network traffic. The last OS utilized is the Windows 7 OS. This system was only used as an administrative console, and as the installation OS for Cauldron.

One of the more interesting methods deployed within DEXTAR is a multi-layered network system approach. The first layer was comprised solely of the scenario virtual machines totaling to 60 VMs. Within this layer, a second layer reserved for participants maintained the IDS system, and the “IT support consoles” that participants utilized within the scenario as their workstation. These first two layers are “scan-able” and are capable of being seen within the network. On top of these two layers is an administrative layer that is effectively invisible to the scenario network, but interacts with it. This administrative layer is responsible for network packet generation, network captures, and other installed metrics. Lastly, the management network surrounds these networks and externally controls those internal networks through VMware API systems only. This allows for the internal network to be “air-gapped”, while having redundant controls.

## 4.4 Cyber Threats

One of the most important aspects about DEXTAR is its ability to provide real world threats. This is created using the BackTrack 5 operating system which as been developed for penetration testing of networks by system administrators. These threats are not modified in any way and are the product of open source development. This development framework is called MetaSploit and is a product of a project with the same name (http://www.metasploit.com). Within the scenario developed, two cyber threats were utilized to exploit machines. In order to add a layer of security, legacy exploits were selected that have since been patched and would not cause any damage to modern computers.

Within DEXTAR, a library of over 1,000 exploitations and vulnerabilities is maintained within the aforementioned metasploit framework. Although the full functionality of this threat generation system is not utilized currently within DEXTAR, further development of scenarios can incorporate more components of this database.

## 4.5 Operation

The operation of DEXTAR requires a systematic approach to ensure that each usage of the DEXTAR test-bed is repeatable between teams. This is accomplished through the use of control scripts, and management practices. As an example, we will go through a startup, scenario run, reset, and shutdown sequence.

Once the DEXTAR servers complete their boot up sequence and are responding to external commands, the following methods are used to administer a scenario run, a reset, and a shutdown sequence:

1. A power on script is used to turn on all relevant virtual machines that are responsible for the running of the scenario. This includes those within the scenario and those acting in a control or administrative function. This script is activated through a PowerCLI (Power Command Line Interface) prompt.
2. Once powered on, the scenario virtual machines are scenario ready based on a previous configuration file.
3. The scenario can be administered in the following fashion (and is the partial script for running the scenario developed):
    a. Clear all SNORT logs and Wireshark logs.

b. Provide participant instructions.
c. Start audio and visual recording.
d. Initiate Ostinato Network Packet Generator.
e. Initiate SNORT and WinIDS.
f. Begin Wireshark Captures on all machines
   i. Attacker
   ii. IDS
   iii. Participant Station
   iv. Network
g. Conduct a three-way ping to synchronize all wireshark logs.
h. Open the console for BackTrack 5
i. Login, and start the GUI.
j. Start Armitage.
k. Conduct a Scan → Quick Scan + OS Detect on 10.160.60.3-6
l. Once 10.160.60.3 and 10.160.60.4 are found, exploit 10.160.60.4 with Dscep.
m. Migrate access.
n. Initiate Pivoting.
o. Exploit 10.160.60.3 while setting LHOST 10.160.60.4.
p. Migrate access.
q. Open Command Shell on 10.160.60.3 and run ipconfig /a
r. Use MSF Scan to scan 10.140.40.3.
s. Exploit 10.140.40.3 from LHOST 10.160.60.3.
t. Migrate access.
u. Explore files and navigate to My Documents and find Server Credentials.txt.
v. Download file and open.
w. Ping AD Server at 10.130.30.5.
x. End activity/scenario.
y. Save Wireshark Captures.
z. Save SNORT logs.
aa. Save MySQL databases.

4. After the completion of the scenario, the reversion script is run. This script resets each virtual machine involved in the scenario, and thereby possibly damaged and currently infected with a virus, to a pre-scenario snapshot previously attained. This allows for an entire network reset and a fresh start for the next team.
5. In order to reduce power consumption, a power off script is implemented that turns off all of the scenario-based virtual machines.
6. At this time, the servers may be powered down through the vCenter power settings if required. Otherwise, they are left in a low power mode.

## 4.6 Data Collection

DEXTAR is designed to collect massive amounts of data in order to identify participant actions, network actions, performance metrics, collaboration metrics, and communication metrics. This is done through a variety of systems.

Internally, DEXTAR uses Wireshark Network Packet Capture programs at each participant terminal, internal and scenario IDS, attacker machine, and hidden installations through out the scenario network. This allows DEXTAR to capture all network traffic with the ability to create a single database containing every single packet from the scenario. The majority of the ground truth can be attained through the collection of the network traffic from the attacker machine. In this instance, wireshark is directly placed onto the attacker machine to provide the cleanest picture of outbound traffic. However, the combination of databases is required in order to identify the entire ground truth. A process that currently while not automated in Phase I, is effective.

Directly interacting with participants, the reporting software captures all IDS alerts that have been classified as either malicious or benign by participants. For each malicious alert, participants are instructed to provide justification for this assessment. Following this, a post-scenario questionnaire is administered asking each participant to provide more information on their perception and comprehension of the scenario they just completed. This data collection aids in determining scores for situation awareness. To boast this even further, communication logs and collaboration logs are combined with this score to provide a more complete picture of both individual and team situation awareness.

The classification of alerts provides an automated algorithm with the necessary information to provide a performance score. This score, a signal detection score, rates the individual or team on their ability to accurately detect and classify alerts.

Lastly, physical methods of data collection are implemented through communication capture devices. Temple mounted microphones record any communication uttered within the scenario. Video cameras capture any movement of physical interaction that team members may have. Currently a post-hoc analytical tool, these recordings are analyzed interaction time, talk time, utterance length, and collaboration.

## 4.7 User Tools

The DEXTAR environment allows for an almost any participant user tool to be installed. Although specific considerations may be required for specific tools, in general, any tool is capable of being run provided the tool is compatible with the implanted software. Currently, the Cauldron and Zenmap tools are installed as the current set of participant tools. These tools both visualize the network, but in vastly differing methods. However, it is common practice to have access to a wireshark system as well as an intrusion detection system. Therefore, these systems have been installed within the DEXTAR system.

## 4.8 Security Protocols

DEXTAR requires the use, and maintenance of a few sensitive systems. Two such systems are BackTrack 5, a penetration testing OS with a library of viruses and exploits, and Cauldron, a restricted access program. In order to provide security several safeguards have been set up. First and foremost, a fair amount of passwords of varying lengths, complexities, and content have all been utilized. Root and Administrative access privileges utilize passwords to the ESXi hosts or VMware that are not implemented within the test-bed in any other location. For BackTrack VMs, not only is access restricted to these machines, with participants never seeing that these exist, a 26 character password was selected to prevent unauthorized access.

A second level of protection was the assignment of the BackTrack OSes to the scenario network. The scenario level network has no access to the internet and is in essence an "air-gapped" system. No service routes, network routes, or links were maintained between the scenario network and the physical network that connected the test-bed together. Only two administrative virtual machines would switch between the two networks – and were required to do so. In these instances, these machines would never be connected to both networks simultaneously.

Lastly, access to the laboratory space was restricted and physical access to the computers was limited. The laboratory was kept locked with the only access granted to those with swipe card access. The computers within the test-bed, aside from the servers, had USB ports shut down to prevent unauthorized usage of USB devices.

## 4.9 Lessons Learned

Throughout the construction of DEXTAR, a constant learning environment was established. It was not simply enough to have a functioning network that would be suitable for implementation within an organization, but this network also had to have several layers of metric and network capture methods. Through trial and error we devised methods for successfully administrating these functions. However, these functions came at the cost of processor power. Future developments within DEXTAR will be further researched for more advanced methods of conducting these metrics with lowered overhead.

Although every attempt was made to not re-invent the wheel, it appears that we may have. Several methods of data collection that were created have commercially available programs that may either increase the capability of the data collection or at the very least streamline the process. In future developments, more time will be taken to examine existing tools and evaluate them for utilization before creating a customized system.

A remaining concern with this implementation of the DEXTAR system is the internal virtual network. Originally it was decided that the network would remain functionally flat in order to allow for a higher level of customization. However, through development and the utilization of network tools, the more this decision proved to be unwise. Future developments of DEXTAR will utilize more networking tools and configurations that are available on higher platforms of VMware in order to get around this concern.

## 4.10 Completed Test-Bed

The completed test-bed is shown in Figure 6. In the foreground are the two experimenter stations. The six participant workstations are seen beyond the experimenter stations.

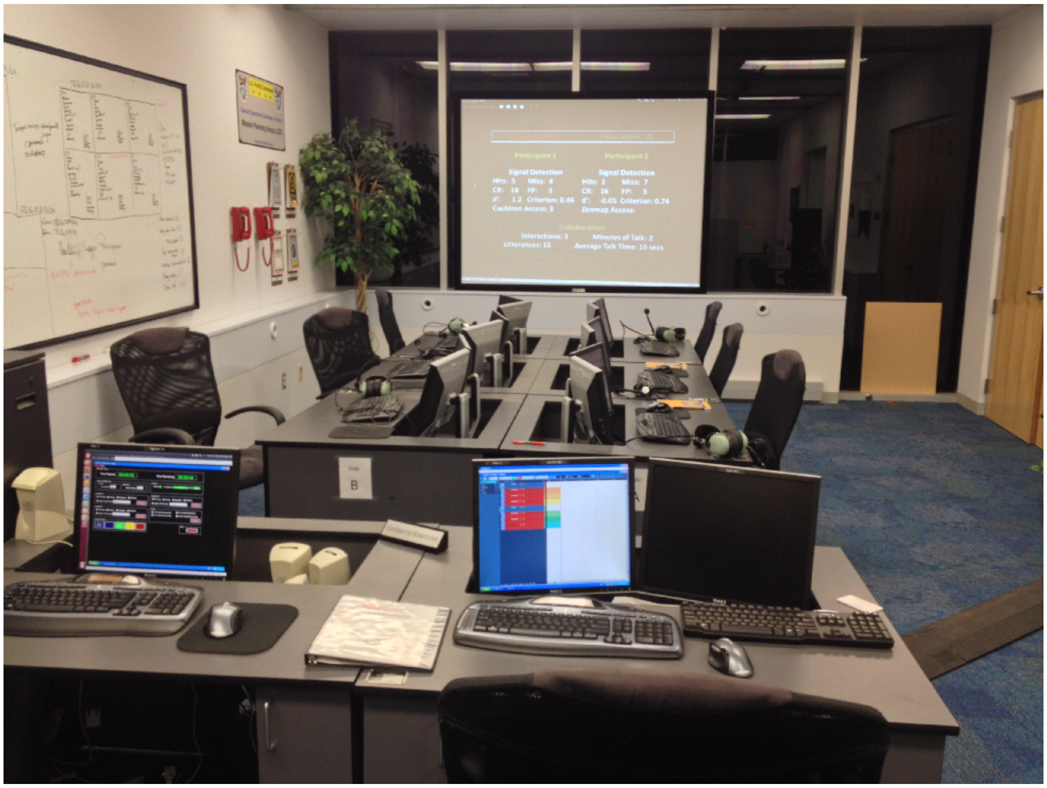


Figure 6. Completed DEXTAR Test-Bed.

## 4.11 References

Alperovitch, D. (2011). Revealed: Operation Shady RAT. *White Paper*, 1–14.

Byres, E., & Lowe, J. (2004). The myths and facts behind cyber security risks for industrial control systems. *Proceedings of the VDE Kongress*, *116*.

Champion, M., Rajivan, P., Cooke, N. J., & Jariwala, S. (2012). Team-Based Cyber Defense Analysis. Presented at the 2012 IEEE International Multi-Dsciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support, New Orleans, LA.

Cooke, N. J., & Shope, S. M. (2004). Designing a synthetic task environment. *Scaled worlds:*

*Development, validation, and application*, 263–278.

D'Amico, A., Whitley, K., Tesone, D., O'Brien, B., & Roth, E. (2005). Achieving Cyber Defense Situational Awareness: A cognitive task analysis of information assurance analysts. *Proceedings of the Human Factors and Ergonomics Society 49th Annual Meeting - 2005*, *2005*, 229–233.

Garrison, C. P., & Ncube, M. (2011). A longitudinal analysis of data breaches. *Information Management & Computer Security*, *19*(4), 216–230. doi:10.1108/09685221111173049

Jariwala, S., Champion, M., Rajivan, P., & Cooke, N. J. (2012). Influence of Team Communication and Coordination on the Performance of Teams at the iCTF Competition, 1–6. doi:10.1177/1071181312561044Not

McNeese, M., Cooke, N. J., & Champion, M. (2011). *Situating Cyber Situation Awareness* (Vol. 10). Proceedings of the 10th International Conference on Naturalistic Decision Making (NDM 2011).

NSA. (2011). CDX 2011 Cyber Defense Exercise, 1–12.

Rajivan, P., Champion, M., Cooke, N. J., Jariwala, S., Dube, G., & Buchanan, V. (2013). Effects of Teamwork versus Group Work on Signal Detection in Cyber Defense Teams. *Foundations of Augmented Cognition*.

Venkatanarayanan, S. (2010, December 12). CyberCog Simulator.

Verizon. (2012). *2012 Data Breach Investigations Report* (pp. 1–80).

Verizon. (2013). *2013 Data Breach Investigations Report*.

Verizon, Politie, & Service, U. S. S. (2011). *2011 Data Breach Investigations Report* (pp. 1–74).

# 5.0 DEXTAR Cyber Analysis Tools

Cauldron maps paths of vulnerability through networks. This lets analysts make optimal network hardening decisions that eliminate attack paths and minimize remediation efforts. It also supports inexpensive what-if analysis for understanding the security impact of proposed network configuration changes.

Cauldron explores the total security ramifications of vulnerabilities accessible to an attacker. The approach follows actual attack patterns, in which a series of exploits incrementally diminishes network security. The attack paths discovered in Cauldron show the exact steps that attackers can take to reach their goals, giving a full picture of network vulnerability. Such complete analysis promotes the development of well-informed security policies. Cauldron analysis is performed off-line, leaving operational networks undisturbed.
In Cauldron, network modeling captures the configuration of the network under analysis. The network model can be populated either manually or from network scans. Cauldron also relies on exploit modeling, in which attacker exploits are modeled in terms of their pre-conditions and post-conditions.

## 5.1 Technical Approach

Cyber security processes are often ineffective against sophisticated multi-step attacks that exploit combined vulnerabilities. Cauldron's integrates a variety of data sources, transforming raw security data into a visual roadmap that delivers a comprehensive network attack model mapping all possible multi-step combined network vulnerability paths.

Cauldron addresses some of the most pressing issues in cyber security, including recognizing real threats, understanding their potential impact on missions, and responding quickly and accurately for minimizing the impact. Capabilities for “what next” and “what if” analysis should be key components of the cyber intelligence capacity of every organization. Visualization of attack paths coupled with Cauldron's modular architecture create a powerful tool for modeling and scenario analysis to gain greater insights about vulnerability management, mission-critical data protection, and advanced cyber intelligence.

Attackers can combine vulnerabilities in unexpected ways, creating numerous options for penetrating a network and compromising mission-critical systems. Organizations often tolerate seemingly low-level vulnerabilities, not realizing that they are stepping stones to more serious consequences. Cauldron exposes vulnerabilities and their interactions, through the combined effect of multiple firewalls, in arbitrary serial and/or parallel topologies. This provides a substantially more robust defense-in-depth methodology.

Expanding threats and increased sophistication of attacks make comprehensive remediation an ever-growing line item in information security budgets. The management challenge is to direct remediation efforts to achieve the most effective overall result, while reining in costs. Cauldron has the unique ability to pinpoint the most effective use of mitigation resources. Cauldron enables remediation teams to quickly find the critical problems to maximize their efforts in securing the enterprise.

Topological vulnerability analysis is the modeling of multi-step attack vulnerability, and then analyzing and visualizing the resulting attack graph (set of all vulnerability paths). Cauldron provides unique capabilities, transforming raw security data into a roadmap that lets you proactively prepare for attacks and manage vulnerability risks.

Cauldron attack graphs provide a common operating picture and a concrete understanding of how individual and combined vulnerabilities impact overall network security. Cauldron integrates a variety of data sources, correlating and normalizing to a common operational model. The data sources include information about the monitored network environment (vulnerability scans, firewall settings, intrusion alerts, etc.), and reported cyber vulnerabilities from a number of sources.

## 5.2 The Cauldron Suite of Cyber Tools

The Cauldron Suite includes a number of tools for building network vulnerability models for analysis within the main Cauldron tool[1].

- **Cauldron**: Builds, analyzes, and visualizes attack graphs according to user-specified scenarios.
- **Importer**: Builds input network models for Cauldron from the outputs of a number of vulnerability scanning tools and firewall devices.
- **Firewall Model Builder**: Builds a Cauldron input model from firewall configurations alone, independent of vulnerability scans, for graph analysis of firewall rule dependencies.
- **Topology Builder**: Supports interactive graphical building of a Cauldron multiple-firewall topology.
- **Service**: Provides interfaces for building Cauldron models from either the command line or from custom Java code.
- **Cisco ASA Parser**: Builds normalized vendor-neutral firewall rule specifications from Cisco ASA and PIX firewall device configurations, for stand-alone analysis or correlation with vulnerability scan imports.
- **Cisco IOS Parser**: Builds normalized vendor-neutral firewall rule specifications from Cisco IOS device configurations, for stand-alone analysis or correlation with vulnerability scan imports.

[1] The Cauldron Suite is installed in the CyVision Program Group under Windows.

- **McAfee FE Parser**: Builds normalized vendor-neutral firewall rule specifications from McAfee FE (formerly called Sidewinder) device configurations, for stand-alone analysis or correlation with vulnerability scan imports.
- **Fortinet Parser**: Builds normalized vendor-neutral firewall rule specifications from Fortinet device configurations, for stand-alone analysis or correlation with vulnerability scan imports.
- **Juniper Parser**: Builds normalized vendor-neutral firewall rule specifications from Juniper device configurations, for stand-alone analysis or correlation with vulnerability scan imports. This tool is currently a beta version.
- **Report Generator**: Creates reports for Cauldron attack graphs (currently, executive summary report).
- **Extras:** Contains user documentation and sample data files.

## 5.3 The Cauldron Architecture

**Error! Reference source not found.** shows the architecture of Cauldron. It builds a model of the network in terms of relevant security data. The vulnerability database is a comprehensive repository of reported vulnerabilities listing the affected software components. Cauldron integrates with network scanners such as Nessus, Retina, Foundstone, nCircle, Core Impact, Qualys, SAINT, nmap, etc. for populating its network model. It imports firewall configuration data to capture policy rules that control access to potentially vulnerable host services.

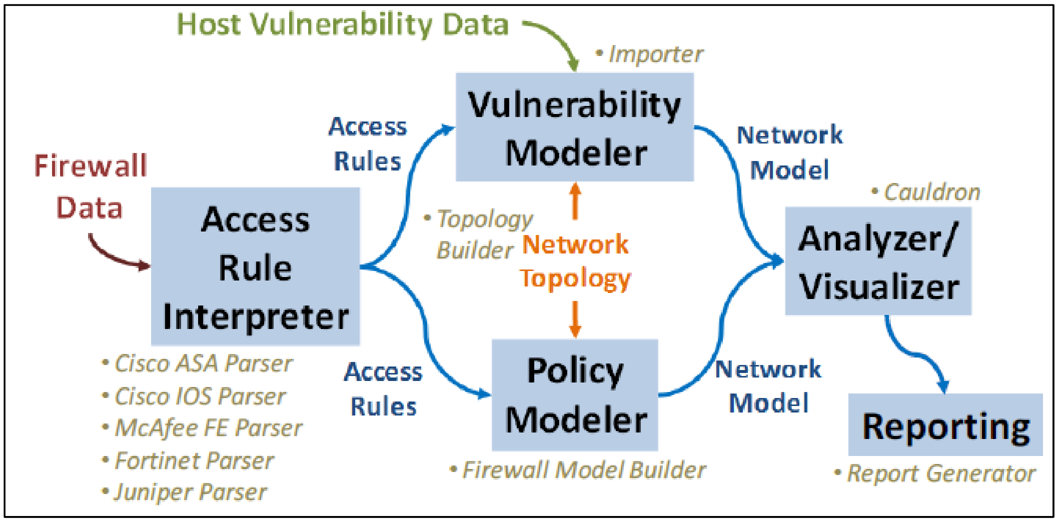


**Figure 7. Cauldron Suite data flows**

Figure 7Figure 7. shows the various possible data flows through the Cauldron Suite. Cauldron's input network models are built via the Vulnerability Modeler (Importer) or Policy Modeler (Firewall Model Builder). Both tools create a normalized correlated dataset for the Analyzer/Visualizer (Cauldron). The Importer and Firewall Model Builder have separate data flows:

- The Importer builds models from vulnerability scans (and optionally, from firewall rules). The resulting Cauldron graph shows network hosts and how they access each other's vulnerabilities.
- The Firewall Model Builder builds models from a set of firewall rules alone. The resulting graph in Cauldron shows network sources/destinations (IP addresses, masks, protocols, and ports) and how they are related through firewall access rules.

In Figure 7, Access Rule Interpreter represents a set of tools that create vendor-neutral firewall access rules from vendor-specific firewall device configurations. These normalized rules provide input to the Firewall Model Builder or (optionally) the Importer. The Service tool lets analysts build Cauldron models from either the command line or from custom Java code. Functionally, it corresponds to the Import tool and the analysis (but not visualization) portion of the main Cauldron tool. The Service tool also supports a data format that unifies the host vulnerability data, network topology, and access rules. This data format is normalized across all supported data sources (vendor neutral).

## 5.4 The Cauldron Attack Graph

The exploit conditions encode how vulnerabilities can be exploited and the results of their exploitation. Together, all these inputs are used to build an environment model for multi-step attack graph simulation. The graph engine uses the environment model to simulate multi-step attacks through the network, matching exploit pre-conditions and post-conditions. The result is all possible paths through the network for a given attack scenario. Cauldron provides sophisticated capabilities for interactive visual analysis of complex attack graphs with high-level overviews and detail drilldown, as shown in Figure .

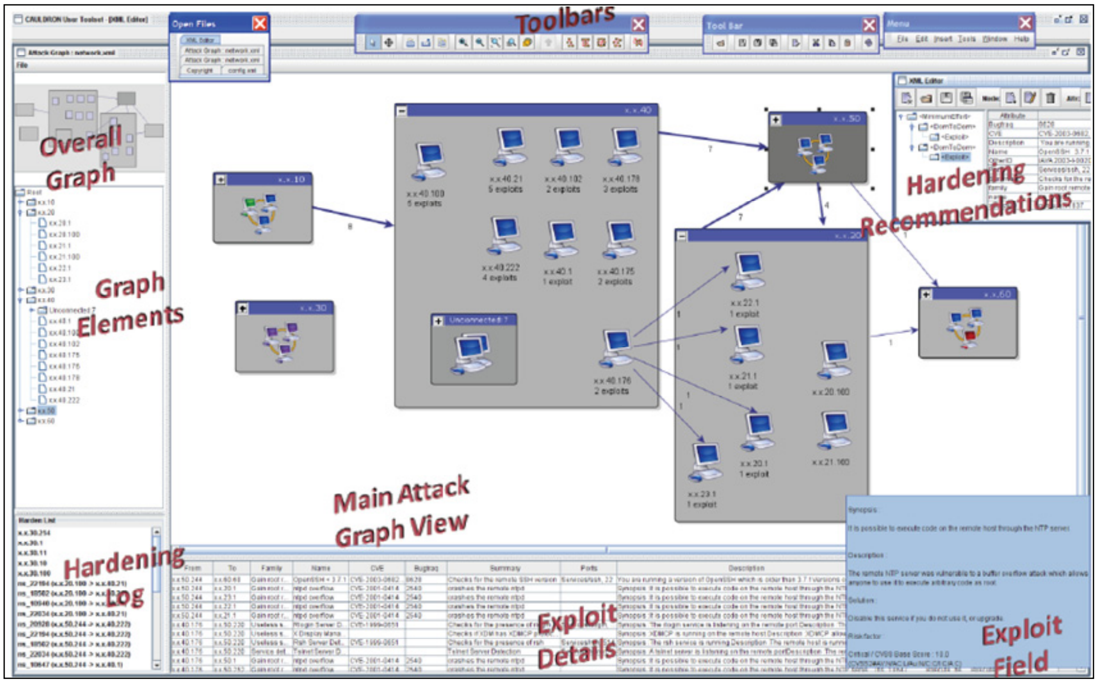


**Figure 8. Cauldron interactive attack graph visualization.**

The Cauldron attack graph aggregates portions of the network into managed zones (e.g., subnets, mission units, etc.) according to its network model. The analyst can begin with a high-level overview of vulnerability paths across zones, and then drill down on demand for interactions among individual hosts, vulnerabilities, etc. Analysts can interactively specify which parts of the network should be hardened (patched, blocked via firewall, etc.). Cauldron also provides recommendations for optimal network hardening.

The Cauldron network model includes detailed data about reported vulnerabilities. There are a number of such vulnerability databases maintained by the government, companies, and the security community. Examples include NIST's National Vulnerability Database (NVD), the Bugtraq security database, Symantec DeepSight, the Open Source Vulnerability Database (OSVDB), and the Common Vulnerabilities and Exposures (CVE) referencing standard.

# 6.0 DEXTAR Metrics and Measures

A benefit of a synthetic task environment like DEXTAR is that not only does it allow participants to exercise skills in cyber analysis, but it also provides an opportunity for measurement. Individual and team performance, process behaviors, and knowledge can be tracked, recorded, and assessed. For instance communication between team members can be observed over time, recorded with time stamps, and analyzed in real time and after the fact to provide indicators of individual and team performance. Resulting metrics can be correlated with performance and diagnostic metrics that are most predictive of performance can be identified. Most importantly, armed with well-validated measures, individuals and teams can be monitored, measured, and the data can be used to intervene to improve individual and team effectiveness, to assess or compare tool effectiveness, to compare training programs or concepts of operation (e.g., team member areas of responsibility) and to better understand cognitive processing in the cyber domain. In the following sections we discuss DEXTAR measures and metrics.

To clarify often confusing terminology, for the purpose of this proposal, we refer to measures as the real-time and cumulative data collected during a simulation or actual session. We refer to metrics as the higher level product that incorporates post-processing and often a range of measures to produce a meaningful assessment of the individual or team. In Phase I we have developed an initial set of measures and metrics. These have been initially validated in the existing CyberCog test-bed using human subjects. In Phase II, we propose additional refinement of the measures and metrics and further experimental validation studies in DEXTAR through human-in-the-loop simulation studies.

## 6.1 Performance

Another benefit of synthetic task environments like DEXTAR is that the space of threats (or ground truth) is known. With this information, the ground truth can be compared to an individual or team's final assessment of a mission. Those that are able to understand the network events in a mission compatible with ground truth will have higher performance scores than those who do not.

Performance points for individual and team performance are earned as follows:

i. Did the user find all of the threats = $\frac{\text{number of threats found}}{\text{number of threats in mission}} \times 100$

ii. Did the user appropriately remediate = $\frac{\text{number appropriate remediations}}{\text{number of threats in mission}} \times 100$

iii. Was mission assurance maintained = yes = 100, no = 0 (partial credit possible)

The performance metric can be calculated at the individual or team level (i.e., how many threats were found by the team as a whole) and is critical because it serves as a criterion measure against which other metrics are validated. We are interested in establishing metrics of process and situation awareness for which changes in the metric correspond to changes in individual or team performance.

## 6.2 Situation Awareness

Situation awareness has been most commonly defined as "the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future" (Endsley, 1988, p. 97). Situation awareness at the individual level is measured by probes inserted within the ongoing task. At the team level situation awareness has often been described as team members "being on the same page." However, for tasks in which there is a true team with heterogeneous members or with very many members, it is not reasonable to assume that all members are on the same page.

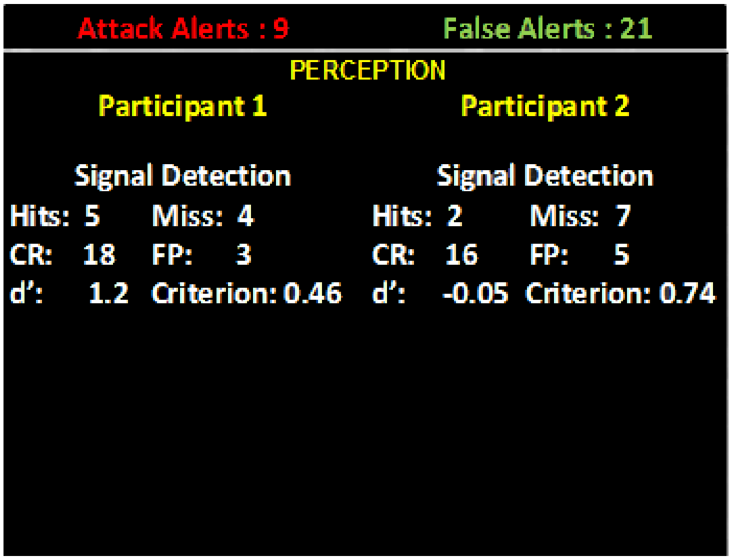
Alternatively, Cooke's approach to the measurement of team situation awareness is ecological in nature (Gorman, Cooke, & Winner, 2006). That is, it relies on a team's ability to identify change in the environment and take action on it. Change in this case could be a new threat on the network or a disruption of service on a system. Time to notice the change can be recorded, as well as whether corrective action was taken. In addition, at the team level, situation awareness entails communication of partial information to the right individuals who jointly construct a model of notion that each team member has to have a full understanding of the situation. Therefore team situation awareness focuses on the efficient coordination of information.

In DEXTAR we measure individual situation awareness in three parts, corresponding to the three parts of the definition: Perception, Comprehension, and Projection. Perception is measured in terms of numbers of alerts processed (per unit time) and of those processed, numbers of hits, false alarms, correct rejections, and misses. Specifically, the goal of the cyber analyst will be to correctly identify threats and to not miss any threats. The detection of threats will be against a background of normal benign network traffic. This is completely analogous to detecting a signal (the threat) in background noise (normal network traffic). The individual situation awareness measures for perception are listed in Table XX. Ultimately these values feed into metrics of sensitivity and decision bias and are appropriate to perception of threat in the triage part of the task.

**Table 3. Perception measures of individual situation awareness.**

| Threat Present | Threat Detected | Signal Detection Performance |
|---|---|---|
| Yes | Yes | Hit |
| Yes | No | Miss |
| No | No | Correct Rejection |
| No | Yes | False Alarm |

Feedback to the experimenter and participants on this measure is presented in Figure 9.

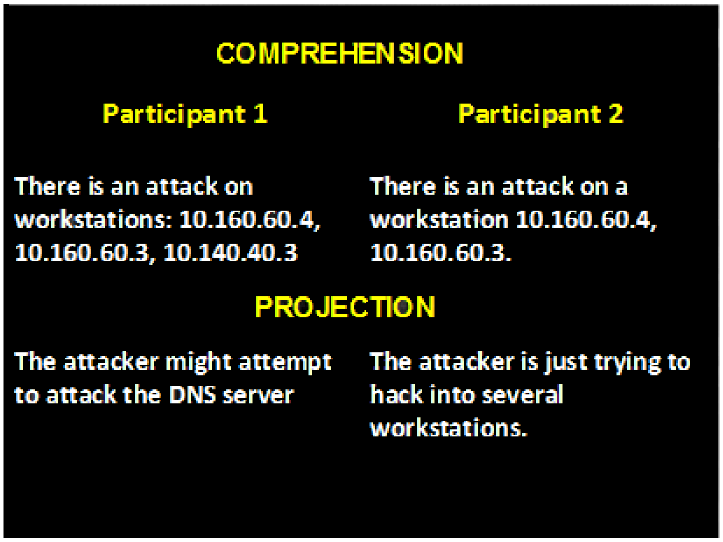


```
      Attack Alerts : 9                    False Alerts : 21
                            PERCEPTION
         Participant 1                        Participant 2

       Signal Detection                     Signal Detection
Hits:  5    Miss:  4               Hits:  2      Miss:  7
CR:   18    FP:    3               CR:   16      FP:    5
d':   1.2   Criterion: 0.46        d':  -0.05    Criterion: 0.74
```

**Figure 9. Feedback on situation awareness perception measure.**

DEXTAR involves analysis beyond triage because the role of a cyber analyst is much more than a cyber threat detection task. Situation Awareness at the Comprehension and Projection levels will involve periodic reports by analysts on not only threat classification, but also their hypothesis about threats to the system and ultimately to the mission (projection). Reported information is recorded as in Figure 10.



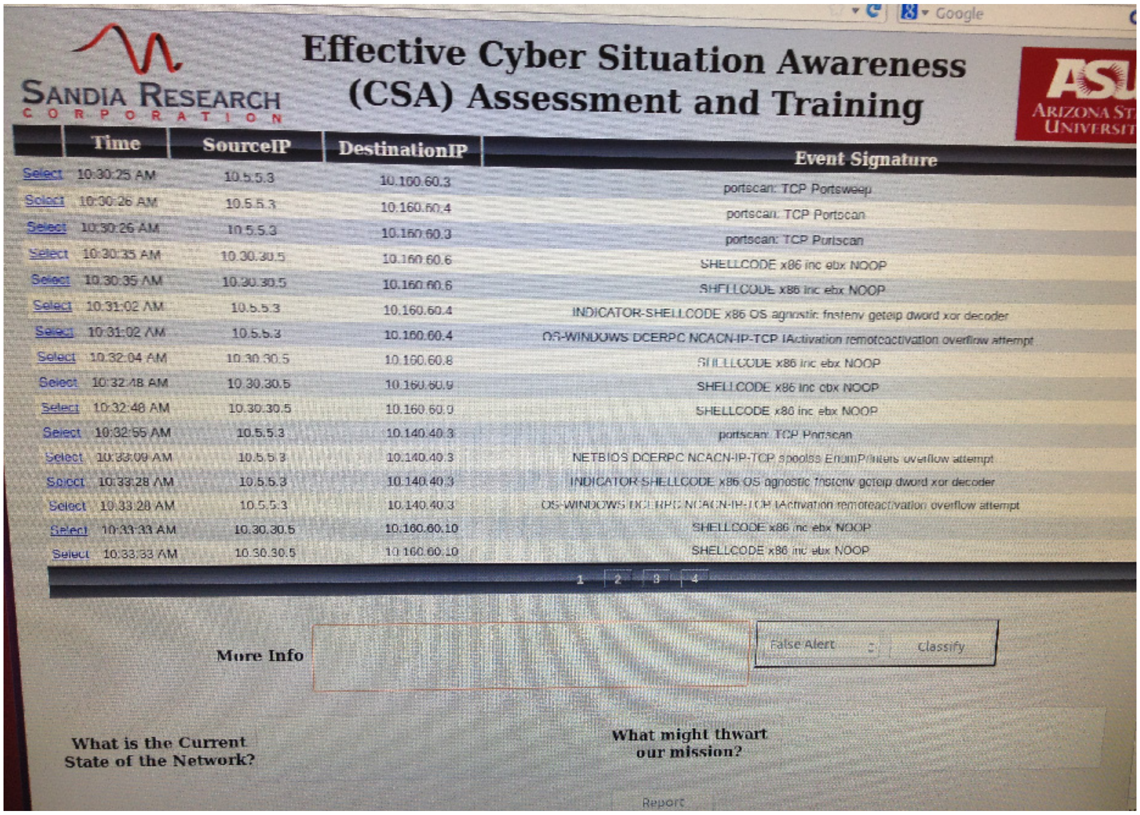
COMPREHENSION

| Participant 1 | Participant 2 |
|---|---|
| There is an attack on workstations: 10.160.60.4, 10.160.60.3, 10.140.40.3 | There is an attack on a workstation 10.160.60.4, 10.160.60.3. |

PROJECTION

| | |
|---|---|
| The attacker might attempt to attack the DNS server | The attacker is just trying to hack into several workstations. |

Figure 10. Report of responses to comprehension and projection probes.



Effective Cyber Situation Awareness (CSA) Assessment and Training

| | Time | SourceIP | DestinationIP | Event Signature |
|---|---|---|---|---|
| Select | 10:30:25 AM | 10.5.5.3 | 10.160.60.3 | portscan: TCP Portsweep |
| Select | 10:30:26 AM | 10.5.5.3 | 10.160.60.4 | portscan: TCP Portscan |
| Select | 10:30:26 AM | 10.5.5.3 | 10.160.60.3 | portscan: TCP Portscan |
| Select | 10:30:35 AM | 10.30.30.5 | 10.160.60.6 | SHELLCODE x86 inc ebx NOOP |
| Select | 10:30:35 AM | 10.30.30.5 | 10.160.60.6 | SHELLCODE x86 inc ebx NOOP |
| Select | 10:31:02 AM | 10.5.5.3 | 10.160.60.4 | INDICATOR-SHELLCODE x86 OS agnostic fnstenv geteip dword xor decoder |
| Select | 10:31:02 AM | 10.5.5.3 | 10.160.60.4 | OS-WINDOWS DCERPC NCACN-IP-TCP IActivation remoteactivation overflow attempt |
| Select | 10:32:04 AM | 10.30.30.5 | 10.160.60.8 | SHELLCODE x86 inc ebx NOOP |
| Select | 10:32:48 AM | 10.30.30.5 | 10.160.60.9 | SHELLCODE x86 inc ebx NOOP |
| Select | 10:32:48 AM | 10.30.30.5 | 10.160.60.9 | SHELLCODE x86 inc ebx NOOP |
| Select | 10:32:55 AM | 10.5.5.3 | 10.140.40.3 | portscan: TCP Portscan |
| Select | 10:33:09 AM | 10.5.5.3 | 10.140.40.3 | NETBIOS DCERPC NCACN-IP-TCP spoolss EnumPrinters overflow attempt |
| Select | 10:33:28 AM | 10.5.5.3 | 10.140.40.3 | INDICATOR-SHELLCODE x86 OS agnostic fnstenv geteip dword xor decoder |
| Select | 10:33:28 AM | 10.5.5.3 | 10.140.40.3 | OS-WINDOWS DCERPC NCACN-IP-TCP IActivation remoteactivation overflow attempt |
| Select | 10:33:33 AM | 10.30.30.5 | 10.160.60.10 | SHELLCODE x86 inc ebx NOOP |
| Select | 10:33:33 AM | 10.30.30.5 | 10.160.60.10 | SHELLCODE x86 inc ebx NOOP |

1 2 3 4
More Info
False Alert
Classify
What is the Current State of the Network?
What might thwart our mission?
Report

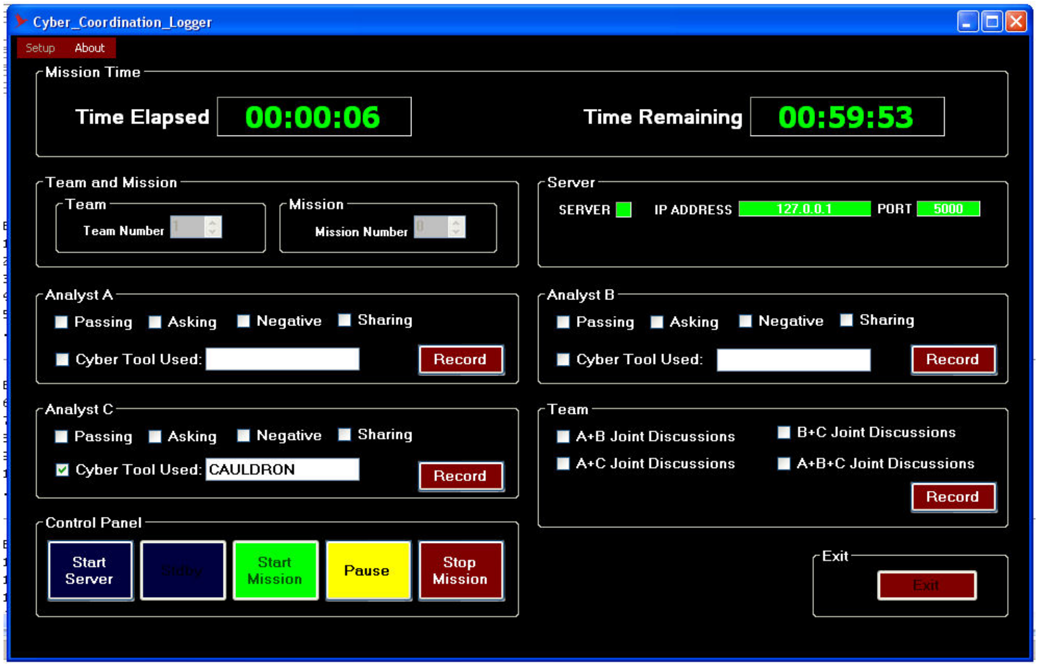
Figure 11. DEXTAR situation awareness logging system

Finally, in Phase II we plan to assess team situation awareness using the CAST (Coordinated Awareness of the Situation by Teams) as reported in Gorman, Cooke, and Winner (2006). This will entail an examination of the coordination on the part of the team to jointly perceive, understand, and mitigate threats.

## 6.3 Process Behaviors

At an individual level process behaviors pertain to individual behaviors in conjunction with the task. We can automatically record certain behaviors such as the numbers of times that a tool is accessed through a keyboard or mouse action. Other individual behaviors such as asking for help are observed and recorded by the experimenter using custom software.

Process behaviors at the team level include communication, coordination, conflict resolution, back-up behavior, and leadership behavior. These behaviors will be observed by an experimenter and recorded with time stamps using custom coordination logging software (see Figure 12). The occurrence of these behaviors will later be analyzed and can be diagnostic with regard to individual or team effectiveness. For instance it may be that those participants who access and use a tool like Cauldron the most have the highest performance scores (see Figure 13).



Cyber_Coordination_Logger

Mission Time

Time Elapsed 00:00:06

Time Remaining 00:59:53

Team and Mission

Team: Team Number 1

Mission: Mission Number 0

Server

SERVER 🟩 IP ADDRESS 127.0.0.1 PORT 5000

Analyst A

□ Passing □ Asking □ Negative □ Sharing

□ Cyber Tool Used:

Analyst B

□ Passing □ Asking □ Negative □ Sharing

□ Cyber Tool Used:

Analyst C

□ Passing □ Asking □ Negative □ Sharing

☑ Cyber Tool Used: CAULDRON

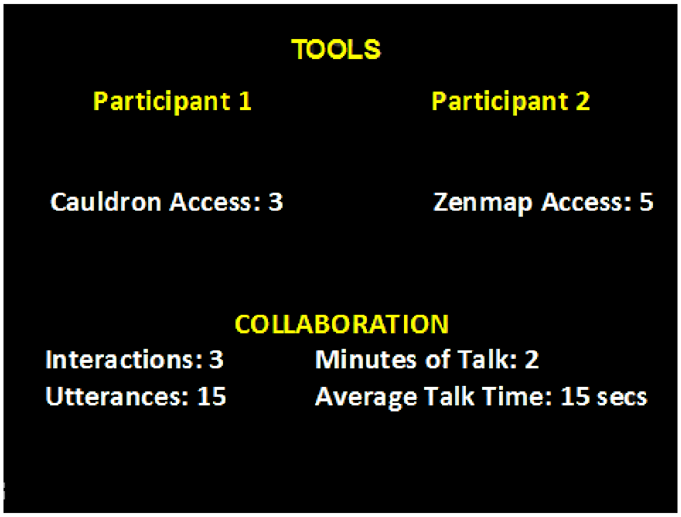
Team

□ A+B Joint Discussions □ B+C Joint Discussions

□ A+C Joint Discussions □ A+B+C Joint Discussions

Figure 12. Cyber Coordination Logger.

Communication (voice or text chat) is a team process behavior that can be directly observed, recoded, and analyzed. In DEXTAR voice and chat are used for participant-to-participant communications. Basic communication measures such as amount of communication can easily be recorded and analyzed. However, communication data provides a rich set of additional metrics including metrics based on communications flow or who talks to whom. In the case of flow, the communications content is not used, but rather a binary time series (or matrix) is used to represent the communications state of a team. A binary “1” indicates a participant is a talker while a binary “0” represents a participant not talking. This metric is especially promising in the cyber domain, given that our cognitive task analyses have identified failures of teamwork as a key limitation (Cooke, Champion, Rajivan, & Jariwala, 2012). Although communication data, consisting of both communication flow and content or context, offer perhaps the richest source of information about team behavior, they are also cumbersome to collect and analyze (Emmert, 1989). Some methods may require up to twenty eight hours per one-hour transcript to analyze for semantic content. Therefore, the automation potential of metrics based on communication flow (i.e., who is talking to whom, when, and for how long) has been a major catalyst of our previous communication flow research. Over the past ten years, we have been conducting research that specifically addresses the relationship between communication flow and team performance and the validation of this relationship (e.g., Cooke, Gorman, Kiekel, Foltz, & Martin, 2005). This work has been successful in identifying communication flow metrics through lag sequential analysis. One finding resulting from these efforts is that communication flow varies significantly as a function of team performance (Kiekel, Cooke, Foltz, Gorman, & Martin, 2002). In DEXTAR the communication that is recorded can be subject to these types of flow metrics or the more laborious content coding methods.



TOOLS

Participant 1 Participant 2

Cauldron Access: 3 Zenmap Access: 5

COLLABORATION

Interactions: 3 Minutes of Talk: 2

Utterances: 15 Average Talk Time: 15 secs

Figure 13. Process feedback.

## 6.4 Other Measures

There are a number of other measures that can be explored and implemented in a Phase II effort. These include measures of individual and team taskwork and teamwork knowledge, perceived workload, and demographic information. The selection of measures and metrics will necessarily depend on the research questions being addressed.

## 6.5 References

Cooke, N. J., Champion, M., Rajivan, P., & Jariwala, S. (2013). Cyber Situation Awareness and Teamwork. *EAI Endorsed Transactions on Security and Safety*. Special Section on: The Cognitive Science of Cyber Defense, 13.

Cooke, N.J., Gorman, J. C., Kiekel, P. A., Foltz, P., & Martin, M. (2005). Using Team Communication to Understand Team Cognition in Distributed vs. Co-Located Mission Environments. Technical Report for ONR Grant no. N00014-03-1-0580

Emmert, P., & Barker, L. L. (Eds.) (1989). *Measurement of Communication Behavior.* White Plains, NY: Longman, Inc.

Endsley, M.R., 1988, Design and evaluation for situation awareness enhancement. In Proceedings of the Human Factors and Ergonomics Society 32nd Annual Meeting, pp. 97–102 (Santa Monica, CA: Human Factors and Ergonomics Society).

Gorman, J.C., Cooke, N. J., & Winner, J.L. (2006). Measuring team situation awareness in decentralized command and control systems. *Ergonomics, 49,* 1312-1325.

Kiekel, P. A.; Cooke, N. J.; Foltz, P. W.; Gorman, J.; & Martin, M. M. (2002). Some promising results of communication-based automatic measures of team cognition. *Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting*.

Jariwala, S., Champion, M., Rajivan, P., & Cooke, N. J. (2012). Influence of team communication and coordination on the performance of teams at the iCTF competition. *Proceedings of the 56th Annual Conference of the Human Factors and Ergonomics Society,* Santa Monica, CA: Human Factors and Ergonomics Society.

Stanton, N.A., Baber, C. and Harris, D. (2008) *Modelling Command and Control: Event Analysis of Systemic Teamwork.* Ashgate: Aldershot, UK.

# 7.0 Scenario and Pilot Testing

## 7.1 Initial DEXTAR Scenario

In the DEXTAR example scenario, Acme is a small start-up apparel company with a small sized internal network. This network is composed of 60 machines split into three sections. The first section is the server group (10.130.30.X) where the company's active directory domain server (main sign on server) and storage server are located, alongside their in-house domain name server. The second section is the network backbone consisting of 9 routers. The last section consists of the three workgroups that end users (workers) would use in order to access the network and conduct the business of the organization. In this scenario, the small business does not host its own website and instead employs an outside firm for this service.

An Intrusion Detection System (IDS) is located within the scenario network. This IDS is utilized by the participants to identify and report on any possible intrusions into the network. This system is supported by 2 participant VMs that access the resources of the IDS and also act as participant workstations within the scenario. These participant VMs can be expanded to include up to 6 participants within the current test-bed.

An external attack source (10.5.5.X) is used by an experimenter to generate an attack on the Acme company network. In this instance, the BackTrack5 OS is used to launch a series of low-level attacks target at specific end computers within the network.

A supportive set of virtual machines assist in creating a realistic network environment to add validity to the scenario. Within the environment 3 packet generators create network traffic that mimics standard internet traffic. This traffic is generated using the Ostinato Packet generator creating packets on the TCP protocols and often points to port 80 or port 8080 on client machines. A secondary set of packet generators, using the NPing program, generate a set of alerts specifically designed to trigger false alerts within the Intrusion Detection Systems available to participants. This is designed to provide a level of noise within the alerts, and to create situations such as a functioning server conducting standard analyses of the network such as scanning, incorrect packet formations, and other rule violations.

Both NPing and Ostinato are customizable in the traffic that is generated. It is possible to control the type of the traffic (TCP, UDP, ARP, ACK, etc.), target and source of traffic (either from or to an IP address, with the ability to specify ports) well as the amount and timing of traffic. This allows us to increase the standard noise available within the network. By adjusting specific settings in Nping we can also adjust the level of false alerts compared to positive alerts to control signal-to-noise ratios. At a minimum, the DEXTAR scenario is set to produce 75% false alerts and 25% positive alerts within the intrusion detection system.

## 7.2 Scenario Background Story

Acme Inc. is an up-and-coming start-up that sells modern apparel. Although this company has taken some precautions within its network configuration and security, it has a limited budget and resources. The company houses several sub-networks with one each for Human Resources, Customer Service, and Production. Each of these networks responds to an in-house Domain server. Acme does not maintain its own production facilities and therefore does not have a SCADA system.

Jason, known as ***C0ff€€ N1nja***, recently received a slightly incorrect order from Acme. Instead of contacting customer care, Jason decides to hack into the network to have a little fun without being too malicious. Jason is known as a script kiddie utilizing pre-fab tools and functions written by other, more experienced hackers, and can only conduct minor infiltrations.

## 7.3 Attack Description

In this scenario the hacker infiltrates the Acme.com system through a single end client computer (10.160.60.4). The hacker first scans this machine using an NMAP scan, and then launches a DS3 vulnerability attack. This attack is effective and quickly takes over the client machine. The attacker then escalates and migrates their privileges creating a stable connection through a spoofed notepad.exe execution. From here, in order to continue the network exploitation without triggering additional IDS alerts, the attacker uses a pivoting table. This essentially routes all traffic from the attacker through the already conquered computer and into the network by using benign traffic methods. Through this pivot table, the attacker scans an additional computer (10.160.60.3) and once again exploits that machine using the same function as before.

At this point, the attacker gathers local information about the second computer that has been exploited. In order to gather this information, the attacker uses a remote command line shell in order to explore and control the computer at an administrative level. The attacker then locates the network information and notes the DNS IP address. Within the scenario, this IP address is set to another client machine and not the actual DNS server (10.140.40.3). The attacker uses a pivot table once again through the second machine, following the first pivot route, and back to the attacking machine. Through this next section, the attacker scans the new IP address (10.140.40.3) and once again infiltrates this machine as with the others before.

Once the third machine has been infiltrated, the attacker examines the computer’s contents. This includes going through files and configuration data. Within the documents folder an administrator account credential file is located. The hacker infiltrates this file and now has administrative login credentials to the Acme server. The hacker is then able to shut down the server using the credentials attained.

## 7.4 Network Diagram

The Acme attack graph is shown in Figure 14.

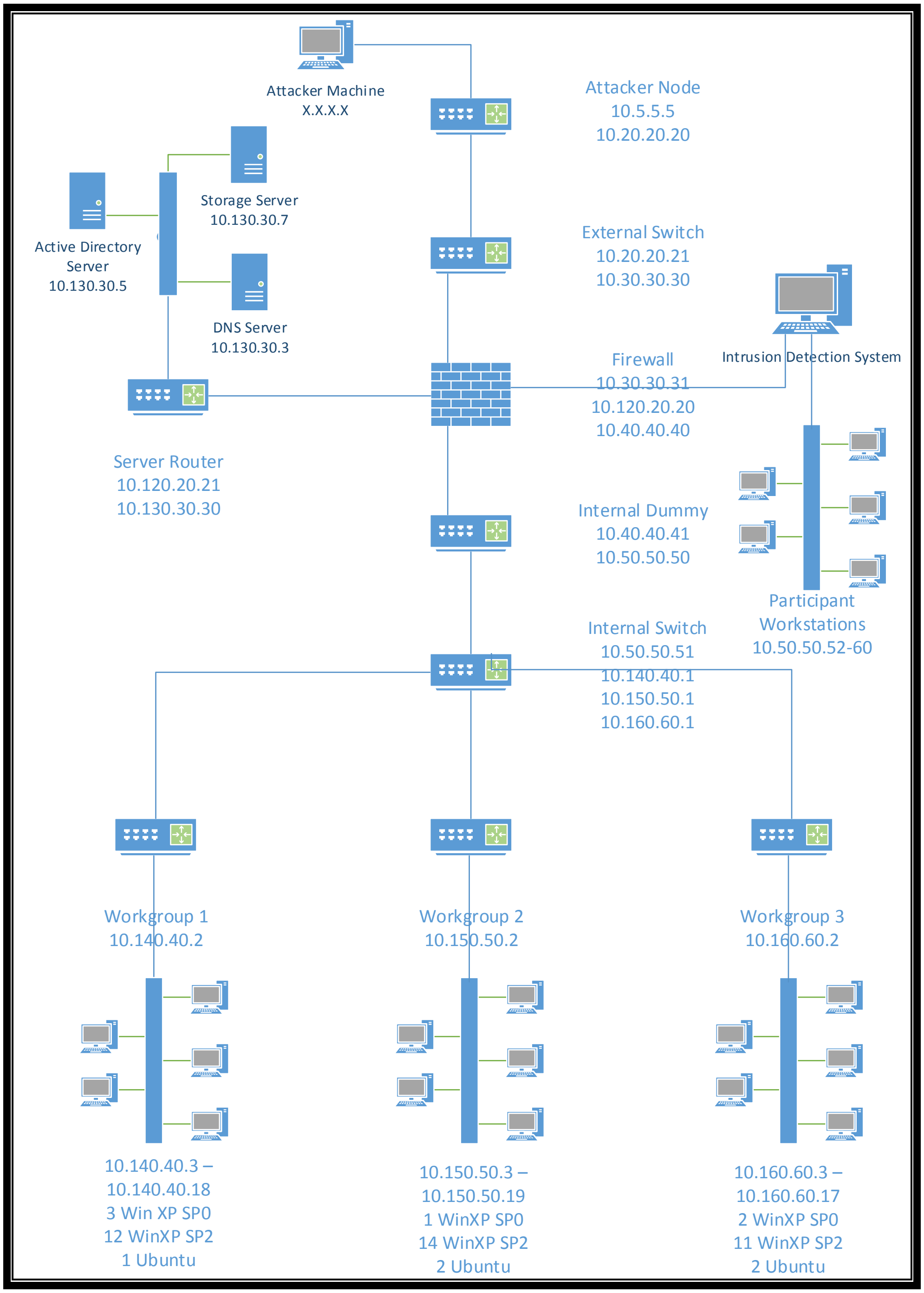


**Figure 14. The Acme company information technology network.**

# 8.0 Summary of Phase I

Our Phase I STTR team is very pleased with the outcome of this project. The feasilbility of a cyber warefare test-bed using validate metrics has been firmly established. In summary these are a few highlights of the Phase I STTR effort:

- Conducted a cognitive needs assessment for the cyber analyst domain
- Designed an effective cyber test-bed
- Constructed the DEXTAR test-bed using a combination of powerful servers and a large array of virtual entities
- Developed a threat injection capability for injecting a wide range of benign and hostile threats
- Custom developed and array of fully functioning metrics and measures
    - real-time
    - post-processing
- Designed an initial scenario
- Implemented this scenario
- Sucessfully pilot tested this scenario with humans-in-the-loop
- Successfully demonstrated this scenario in October 2013 at the MURI Cyber Workshop
- Learned valuable lessons to be applied in larger scale operations in a possible Phase II effort.

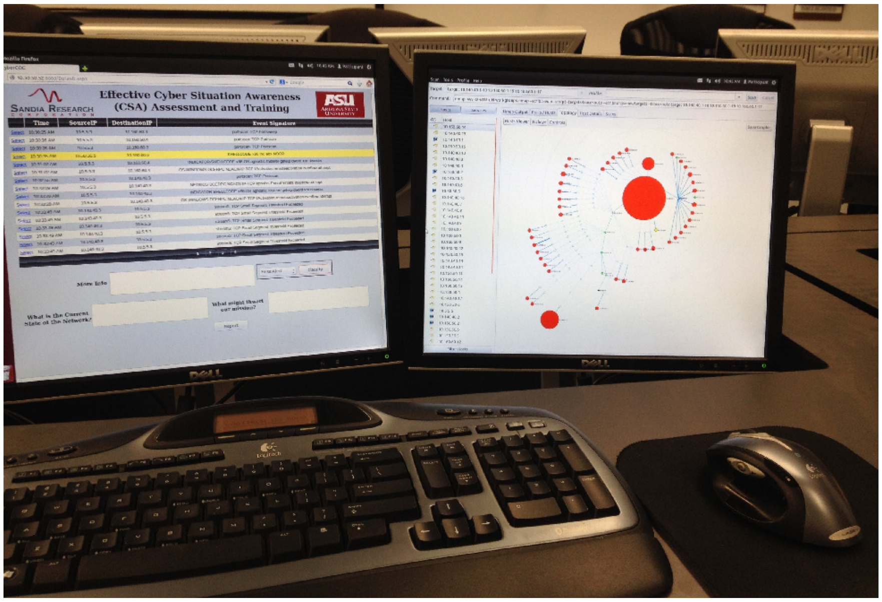

Effective Cyber Situation Awareness
(CSA) Assessment and Training
Sandia Research Corporation
ASU
DELL
DELL

Figure 15. DEXTAR participant workstation.

# 9.0 Summary of Commercial Potential

The Phase I STTR team is excited about the commercialization possibilities that can result from the project. The recent MURI cyber workshop, help in Mesa, AZ highlighted the need for improved cyber security - especially at the human analyst level. It was also noted that **groups** of analysts need to begin functioning as a **teams** of analysts.

We see the following commercialization opportunities resulting from this STTR.

## 9.1 Cyber Tools

Improved cyber tools that address the needs of the analyst are needed. These tools need to be properly validated in a lab such as DEXTAR. These tools are needed at both the individual and at the team level. Also needed are new and effective collaborative tools for cyber analysts including tools to manage groups of threats, assigning threats to individual analysts, and situation awareness sharing tools.

In summary the commercial possibilities include:

- Tool evaluation and assessment
- New tool development
- Tool comparisons

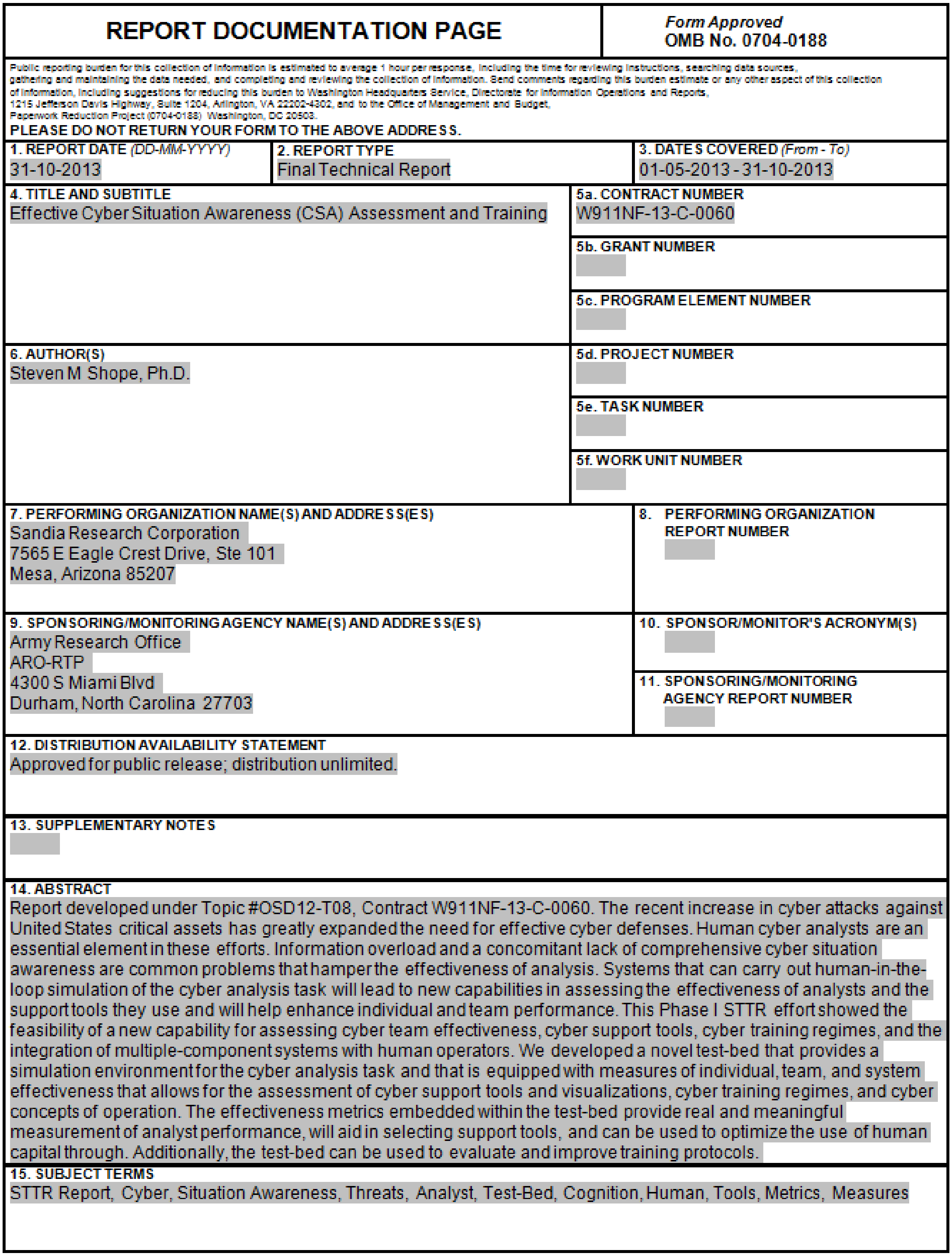
## 10.2 Cyber Training

Improved cyber training is greatly needed. This includes the design and development of training methodologies, training materials, training software and training assessment tools. Again, this training needs to take place at both the individual and team levels.

In summary the commercial possibilities include the development of individual and team

- Training methodologies
- Training materials
- Training software
- Training assessment tools
- Training

# 10.0 SF-298



## REPORT DOCUMENTATION PAGE

*Form Approved*
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Service, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*
31-10-2013

2. REPORT TYPE
Final Technical Report

3. DATES COVERED *(From - To)*
01-05-2013 - 31-10-2013

4. TITLE AND SUBTITLE
Effective Cyber Situation Awareness (CSA) Assessment and Training

5a. CONTRACT NUMBER
W911NF-13-C-0060

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Steven M Shope, Ph.D.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Sandia Research Corporation
7565 E Eagle Crest Drive, Ste 101
Mesa, Arizona 85207

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Army Research Office
ARO-RTP
4300 S Miami Blvd
Durham, North Carolina 27703

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

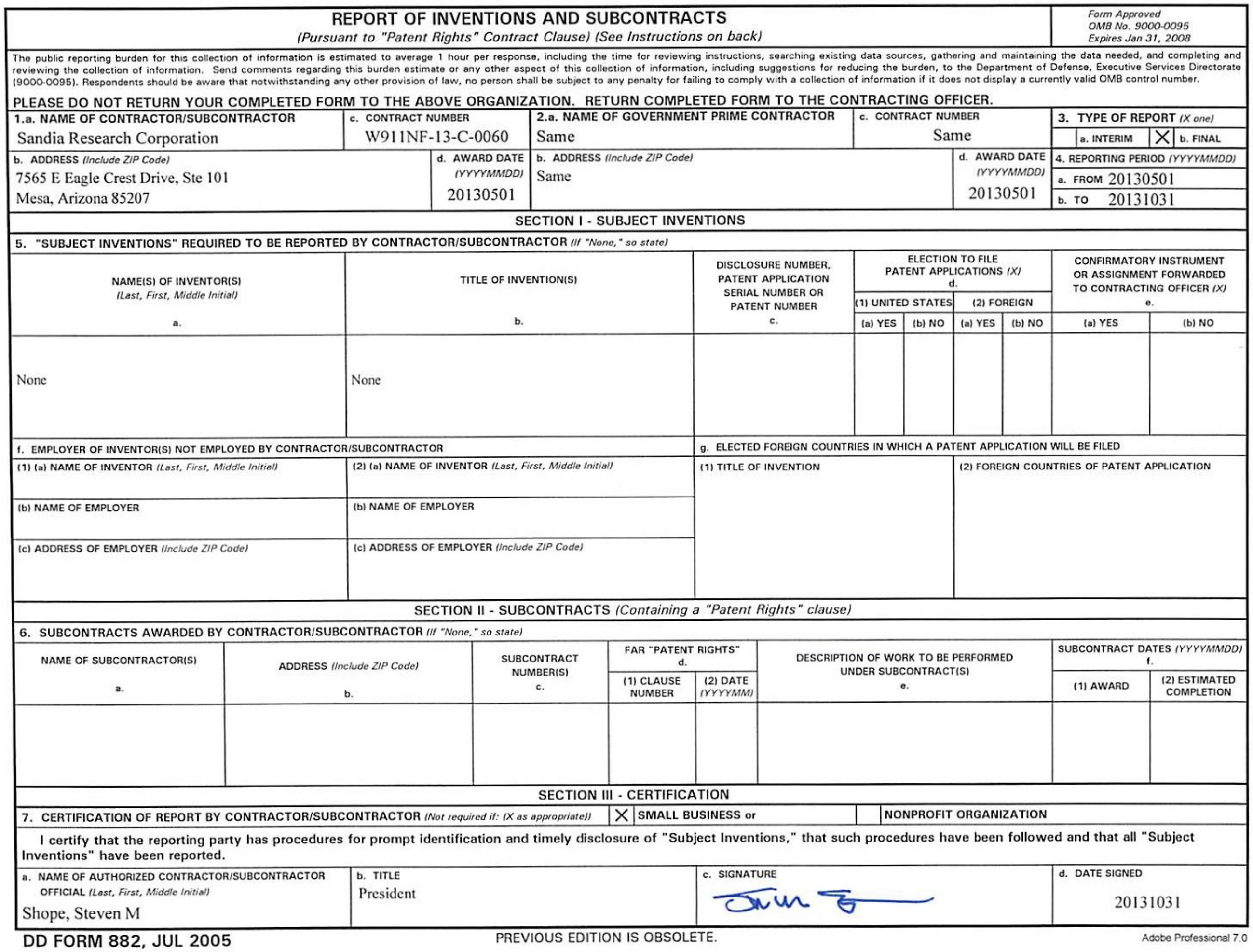
13. SUPPLEMENTARY NOTES

14. ABSTRACT

Report developed under Topic #OSD12-T08, Contract W911NF-13-C-0060. The recent increase in cyber attacks against United States critical assets has greatly expanded the need for effective cyber defenses. Human cyber analysts are an essential element in these efforts. Information overload and a concomitant lack of comprehensive cyber situation awareness are common problems that hamper the effectiveness of analysis. Systems that can carry out human-in-the-loop simulation of the cyber analysis task will lead to new capabilities in assessing the effectiveness of analysts and the support tools they use and will help enhance individual and team performance. This Phase I STTR effort showed the feasibility of a new capability for assessing cyber team effectiveness, cyber support tools, cyber training regimes, and the integration of multiple-component systems with human operators. We developed a novel test-bed that provides a simulation environment for the cyber analysis task and that is equipped with measures of individual, team, and system effectiveness that allows for the assessment of cyber support tools and visualizations, cyber training regimes, and cyber concepts of operation. The effectiveness metrics embedded within the test-bed provide real and meaningful measurement of analyst performance, will aid in selecting support tools, and can be used to optimize the use of human capital through. Additionally, the test-bed can be used to evaluate and improve training protocols.

15. SUBJECT TERMS
STTR Report, Cyber, Situation Awareness, Threats, Analyst, Test-Bed, Cognition, Human, Tools, Metrics, Measures

# 11.0 SF-882



## REPORT OF INVENTIONS AND SUBCONTRACTS

*(Pursuant to "Patent Rights" Contract Clause) (See Instructions on back)*

*Form Approved*
*OMB No. 9000-0095*
*Expires Jan 31, 2008*

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to the Department of Defense, Executive Services Directorate (9000-0095). Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

**PLEASE DO NOT RETURN YOUR COMPLETED FORM TO THE ABOVE ORGANIZATION. RETURN COMPLETED FORM TO THE CONTRACTING OFFICER.**

1.a. NAME OF CONTRACTOR/SUBCONTRACTOR: Sandia Research Corporation

c. CONTRACT NUMBER: W911NF-13-C-0060

b. ADDRESS *(Include ZIP Code)*:
7565 E Eagle Crest Drive, Ste 101
Mesa, Arizona 85207

d. AWARD DATE *(YYYYMMDD)*: 20130501

2.a. NAME OF GOVERNMENT PRIME CONTRACTOR: Same

c. CONTRACT NUMBER: Same

b. ADDRESS *(Include ZIP Code)*: Same

d. AWARD DATE *(YYYYMMDD)*: 20130501

3. TYPE OF REPORT *(X one)*: □ a. INTERIM ☒ b. FINAL

4. REPORTING PERIOD *(YYYYMMDD)*
a. FROM 20130501
b. TO 20131031

### SECTION I - SUBJECT INVENTIONS

5. "SUBJECT INVENTIONS" REQUIRED TO BE REPORTED BY CONTRACTOR/SUBCONTRACTOR *(If "None," so state)*

| NAME(S) OF INVENTOR(S) *(Last, First, Middle Initial)* a. | TITLE OF INVENTION(S) b. | DISCLOSURE NUMBER, PATENT APPLICATION SERIAL NUMBER OR PATENT NUMBER c. | ELECTION TO FILE PATENT APPLICATIONS (X) d. (1) UNITED STATES (a) YES | (1) UNITED STATES (b) NO | (2) FOREIGN (a) YES | (2) FOREIGN (b) NO | CONFIRMATORY INSTRUMENT OR ASSIGNMENT FORWARDED TO CONTRACTING OFFICER (X) e. (a) YES | (b) NO |
|---|---|---|---|---|---|---|---|---|
| None | None | | | | | | | |

f. EMPLOYER OF INVENTOR(S) NOT EMPLOYED BY CONTRACTOR/SUBCONTRACTOR

(1) (a) NAME OF INVENTOR *(Last, First, Middle Initial)*

(b) NAME OF EMPLOYER

(c) ADDRESS OF EMPLOYER *(Include ZIP Code)*

(2) (a) NAME OF INVENTOR *(Last, First, Middle Initial)*

(b) NAME OF EMPLOYER

(c) ADDRESS OF EMPLOYER *(Include ZIP Code)*

g. ELECTED FOREIGN COUNTRIES IN WHICH A PATENT APPLICATION WILL BE FILED

(1) TITLE OF INVENTION

(2) FOREIGN COUNTRIES OF PATENT APPLICATION

### SECTION II - SUBCONTRACTS *(Containing a "Patent Rights" clause)*

6. SUBCONTRACTS AWARDED BY CONTRACTOR/SUBCONTRACTOR *(If "None," so state)*

| NAME OF SUBCONTRACTOR(S) a. | ADDRESS *(Include ZIP Code)* b. | SUBCONTRACT NUMBER(S) c. | FAR "PATENT RIGHTS" d. (1) CLAUSE NUMBER | (2) DATE *(YYYYMM)* | DESCRIPTION OF WORK TO BE PERFORMED UNDER SUBCONTRACT(S) e. | SUBCONTRACT DATES *(YYYYMMDD)* f. (1) AWARD | (2) ESTIMATED COMPLETION |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

### SECTION III - CERTIFICATION

7. CERTIFICATION OF REPORT BY CONTRACTOR/SUBCONTRACTOR *(Not required if: (X as appropriate))* ☒ SMALL BUSINESS or □ NONPROFIT ORGANIZATION

I certify that the reporting party has procedures for prompt identification and timely disclosure of "Subject Inventions," that such procedures have been followed and that all "Subject Inventions" have been reported.

a. NAME OF AUTHORIZED CONTRACTOR/SUBCONTRACTOR OFFICIAL *(Last, First, Middle Initial)*: Shope, Steven M

b. TITLE: President

c. SIGNATURE:

d. DATE SIGNED: 20131031

DD FORM 882, JUL 2005 PREVIOUS EDITION IS OBSOLETE. Adobe Professional 7.0

# Appendix

# DEXTAR Operations Manual

# Cyber Defense Exercises for Team Awareness Research

# D.E.X.T.A.R

Manual of Operations

A project sponsored by:
The Army Research Office
Cliff Wang, Liaison

Built in partnership between:

# Table of Contents

# Tables Index

# Figures Index

## Acknowledgements

We would like to thank the Army Research Office for the support and sponsorship of this project. We would like to extend heartfelt gratitude to Cliff Wang. Without his guidance and support, this project would not be possible. We would like to also thank all of those whom have provided information, support, and feedback through this process.

## DEXTAR Project Team

***Sandia Research Corporation***
Dr. Steven Shope, Principle Investigator
Stephanie Lambert, Administrator
Kyle Uithoven, Programmer

***Arizona State University***
Dr. Nancy Cooke, Co-Principle Investigator
Michael Champion, System Architect and Administrator
Prashanth Rajivan, System Architect and Programmer

***George Mason University***
Dr. Sushil Jajodia, Co-Principle Investigator
Dr. Massimiliano Albanese, Cauldron Support

# Introduction

The DEXTAR project is the culmination of several years-worth of work dedicated to understanding cyber situation awareness and human performance within cyber defensive positions. The foundational work for this project was started with a program called CyberCog. This project was dedicated to understanding the decision making and awareness of triage level cyber analysts. This level was not meant to examine or solve an issue, but to simply identify an area of interest. However, this project began to show limitations as the fidelity of the system was relatively low, and while in abstract showed ecological validity, the practical function of it was not.

While the Cyber team at the CERTT lab at ASU continued to examine the possibilities of CyberCog and how it can be expanded, additional research went underway to identify more areas of interest within cyber defense personnel. This led to the observation of several opponent-based cyber games between military institutions, collegiate institutions, and individual level situations. From these, the foundation of DEXTAR was formed by repurposing the infrastructures found within each of these scenarios. A considerable amount of research went into examining how to effectively augment or alter these constructs into human performance testing parameters that CyberCog had previously attempted.

The research undertaken was carried out from a variety of sources with a variety of goals. The goal of this test-bed has been to increase our knowledge of the cyber security domain and the practices of those within it. Through sponsorship from the Army Research Office this project was able to develop a prototype. This manual is an overview of the construction of the test-bed as well as minor operations of it.

It should be noted that the operations of this test-bed are not bound or artificially limited by what is within this manual. Instead, the system is only bound by its own physical parameters on what it can accomplish. New software, hardware upgrades, and programming can effectively maintain this system and expand its capabilities far beyond the scope of this manual. This manual is simply a description of the environment.

# Test-bed Hardware

The DEXTAR project utilized the existing structure of the MacroCog test-bed located in the CERTT Lab at Arizona State University. This system was developed by Sandia Research Corporation and consists of a number of participant stations, experimenter stations, audio and visual recording, and data recording. This served as the basis for the DEXTAR system and utilized a majority of the same hardware. Below in **Error! Reference source not found.** is a diagram of the MacroCog test-bed.

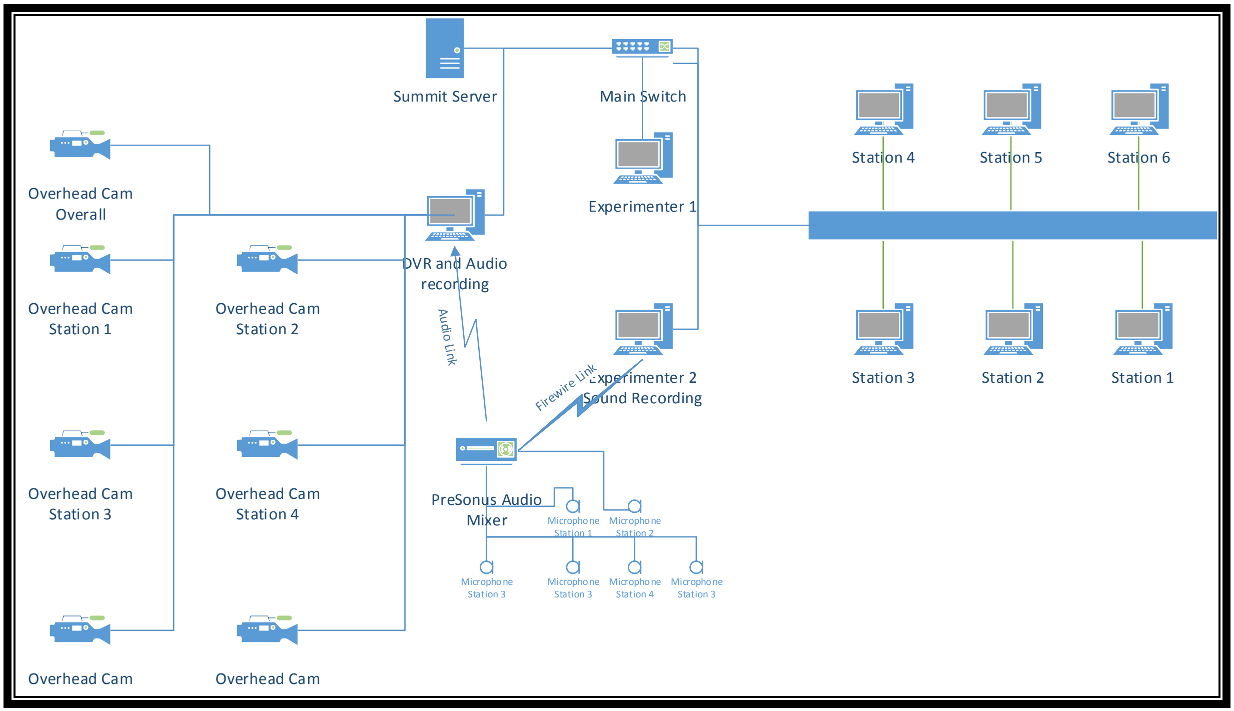


***Figure 1. MacroCog Test-bed Configuration***

In overview, the MacroCog system is comprised of five components to support an experimental environment:

1. Six Participant stations for participants to interact with the environment
2. Two Experimenter positions for the control, monitoring, and maintenance of experiments.
3. An Audio recording system where a microphone is placed at each participant station
4. A video recording system which captures the video from each of the six participant stations. These cameras are cross-mounted overhead from ceiling racks.
5. A server system, called Summit, that controls the experiment software and maintains the environment.

## Pre-existing Computers

The system is comprised of 8 standard computers, 1 DVR system, and 1 server.

The standard computers are as follows:
Dell Optiplex 740
1TB Hard drives
1GB Ram
Logitech MX5000 Bluetooth Desk Set

Specific service tag numbers may be found on each machine. These computers are used for the participant stations and experimenter stations. Each participant computer had its hard drive replaced from the existing 80GB hard drives to a 1TB hard drive to account for extra storage space that may be required from storing data from the scenarios.

The DVR system is a custom built machine containing;
XXGB hard-drive
XXGB RAM
Aventura DVR System of hardware modifications and software control mechanisms.

The Summit server consists of:
Dell PowerEdge 2950
Dual-Core 1.6 GHz
8GB RAM
3TB Total Storage capacity

The Summit server currently runs Windows Storage Server 2003 with several additional components for website hosting, SQL engine and database management, and network management. This machine has two gigabit network interfaces cards, however, one has malfunctioned and is inoperable.

## Audio-Visual System

The test-bed's audio/visual system compromised of two separate systems. The first system is run by the Aventura DVR system and contains 7 overhead cameras. Each camera is a standard ceiling mount camera deriving power from a camera supply unit located inside the rack mount system. Six cameras are cross-mounted and are aimed at each of the participant stations with a separate feed into the DVR system. The seventh camera is an overall view that captures the entire test-bed. The DVR system then combines the audio recorded from all channels into this camera's feed for an overall video of a given scenario/recording.

The audio recording system is split into two parts at the time the test-bed was converted. The original system consisted of six microphone inputs from each of the participant stations into a PreSonus FP10 (24 bit/96k recording interface). This pre-amp was connected the experimenter right station via firewire for audio recording using CuBase with customized templates. A single combined feed was sent into the DVR system through an audio cable. The microphones were temple mounted noise-reducing shure microphones.

The second aspect of the audio system relied on the participant machine's input. Four participant machines (stations 1,3,4, and 6) received David Clark USB headsets while 2 machines (stations 2 and 5) received similar headsets by Logitech. These headsets are attached to each participant machine to allow for certain amounts of participants to interact with each other without having other participants overhear conversations. This is useful in situations where two experiments are being run at once, or two or more instances of the same experiment are running with two or more teams in the test-bed. The temple mounted microphones were then attached to these microphones so that participants were speaking into both systems. Only the temple mounted microphone system would record audio for analytic purposes.

## Network Hardware

The test-bed network was fairly simple in construction. There was a LinksKey 10/100/1000 mbps dummy switch in which all computers were connected. Each computer had a gigabit network interface card along with the server containing one functional gigabit network interface card.

## Uninterrupted Power Supply

The APC UPS system is responsible for providing clean power to each computer within the rack. This system boasts approximately 15 minutes worth of backup power. At this time, only the Summit server is configured to shut down if a power loss is detected by the UPS. This is because only one data connection to the UPS is allowed at any given time. However, the UPS is connect via 25' surge protector to the building's main backup generator power supply (grey plug). Although the system has not been tested, this method should allow continual uninterrupted power to each of the servers. In the event of power loss, the rest of the test bed will lose power instantly. Only the servers are on backup power supply.

## DEXTAR Server Additions

One of the key components of DEXTAR was the addition of two servers with the following specs:
Dell PowerEdge R420
Service Tag 44SSGX1 (TOP) 44RWGX1(BOTTOM)
2TB RAID 1 HD (2x 2TB HD)
Intel Xeon E5-2420 1.9GHz 6-core Hyperthreaded (48 logical cores)
16GB RAM (Deimos/Bottom)
48GB RAM (Phobos/Top)
4x Gigabit NIC each
At the time of installation, the servers were equipped with lesser processors and equal RAM memory. Due to limitations within the system, the processors were upgraded to those listed above. The RAM was then upgraded after preliminary benchmarking tests had accidentally crashed the server due to limited memory space.

The servers have a 2TB RAID 1 Array to ensure safety of data and a certain level of stability. This RAID system does not provide extra speed, but instead provides redundancy within the data storage. These servers have no current backup storage device and must act as their own. If a hard drive fails, please contact a system administrator to resolve the problem.

The second key installation from the DEXTAR test-bed was the addition of a web-managed smart switch:
Dell PowerConnect 2428
Serial: CN0F491K2829832I0414A08
MAC Address: D0:67:E5:C2:F9:FE
Service Tag: F64WTS1
Name: Charon (configured)
Ports: 24 (with two optional fiber inputs for the addition of high speed fiber, but this is not needed for DEXTAR)

This switch was installed just below the MacroCog switch and is meant to run in parallel for the integration of the test-bed. This dual configuration offers two major components:

1. Separation of the MacroCog server environment
2. Compartmentalizing the effects that the DEXTAR environment could have during a catastrophic system failure.
    a. Such effects could be the release of any number of viruses
    b. The permanent damage to network traffic between servers due to high flow of traffic that could overload Summit.

Current Cabinet configuration is as follows in Figure 2:

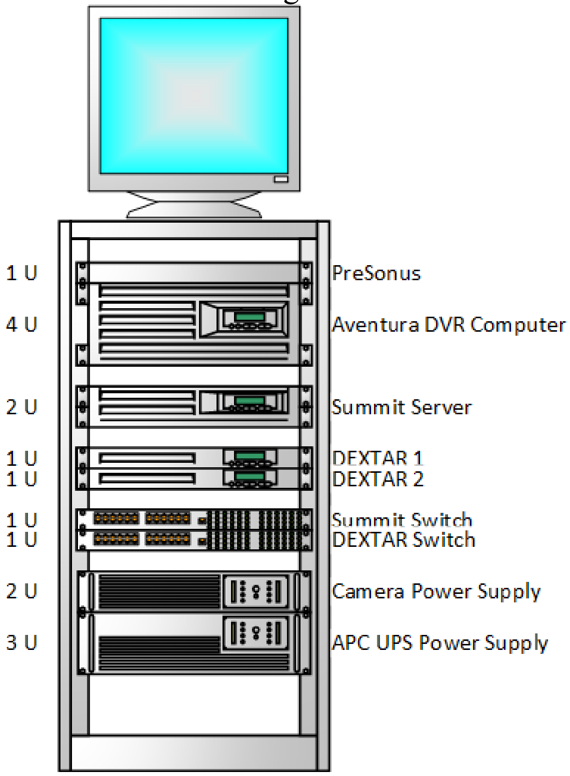


*Figure 2. Current Rack Cabinet Configuration with DEXTAR components*

## Proper Maintenance

The test-bed hardware components of both the Summit and DEXTAR systems need regular maintenance. As these systems exist in Phoenix, AZ there is the unfortunate aspect of dust and sand that can wreak havoc on computer components. These components will need regularly cleaning, and is recommended once a year. To clean these components the following steps will need to be taken:

1. Disconnect all power to each component, and disconnect power cords from each device.
2. Once power is disconnected, press the power button on each component. This will cause any stored power to dissipate and render the device safe for opening.
3. Remove only one device at a time. The following steps will describe the cleaning of servers:
    a. Remove the server from the rack system. The two DEXTAR servers are on minimal rails and the locks will need to be pressed to release each machine from each rail. The Summit Server is on a more elaborate rail system. This system requires a lock to be released, it has a blue handle on it that needs to be pressed in.
    b. Once removed from the rack you will need to most likely take the machine outside, or somewhere in the facility that will be OK if it gets incredibly dusty.
    c. Attach a grounding cord from yourself to the chassis of the computer before removing the cover of the machine. You should also be wearing tennis shoes or thick rubber soled footwear. It is not advisable to do this barefoot for safety reasons.
    d. To remove the cover, you'll need a coin to turn the locking mechanism to unlock. Then pull the tab up. This will move and then lift the cover off of the servers. There should be nothing attached to the cover, but you will need to be careful about the removal.
    e. Once removed you'll have exposed the circuitry of the server, be mindful of this. From here, take canned air and spray through the computer. Avoid spraying into individual components such as hard drive containers, power supplies, and DVD drives. This can potentially cause damage. You should remove the power supply(ies), hard drives, and DVD drives before conducting a cleaning to help reduce risk.
    f. There are a row of fans within the server. You are able to easily remove each fan and should do so to clean them out. This will also open up more area for to clean the dust out.
    g. **DO NOT UNDER ANY CIRCUMSTANCE REMOVE THE HEATSINKS OR PROCESSORS!** This could lead to potentially serious or fatal damage to the system. Do not do it. You may use canned air to blow through heatsinks while attached. You should clean them this way, and this way alone.
    h. Wipe down with an anti-static cloth the inside of the server's cover. This will help reduce excess dust inside the system. Do not use any solutions, sprays, or chemicals.

    i. Wipe down the exterior of the system with an anti-static cloth. Do not use any solutions, sprays, or chemicals. Just a clean anti-static cloth.
    j. Replace the components back as you found them and close the cover. Replace the system when you are done cleaning.
    k. Place the system back on the rail system that it came from and in the position it came from. While technically the machines do not need to be in order, it is ideal for wiring purposes and this manual is based on the aforementioned configuration.
    l. This procedure is used for each server, including Summit. Although summit has minor internal changes from the DEXTAR systems, it is the same principle.

4. The DVR system will likely not need to be disassembled for cleaning. On the front of the DVR system there are a series of filters that block most dust particles. Remove these filters, clean them, and replace them. If the DVR system is in need of a full clean, follow these instructions:
    a. Disconnect all power cables and remove from the rack by removing the bolts holding it into the rack system. You may need to label cables in order to place them back correctly if the labels have fallen off.
    b. Open up the system. This system is far more open than the servers with far less to remove.
    c. Using canned air, blow the dust out of the system and avoid blowing directly into the power supply or DVD drive.
    d. Wipe the cover down with an anti-static cloth. Do not use chemicals, solutions, or sprays. Just a clean anti-static cloth.
    e. Reattach to the rack, and reconnect.
5. There is no purpose to opening the switches to clean them and it is adamantly advised against. Simply wipe the intake/exhaust from dust and wipe the front of the system with an anti-static cloth. This should be completed for the UPS, Camera power supply, and PreSonus.
6. During yearly cleaning, it will be necessary to check all cables to ensure none have degraded. Simply check for physical signs of degradation and wear. If a cable is too worn, replace it. This is especially needed for power cables as a fault within these cables will cause problems.

# Networking

One of the key elements of the DEXTAR system is the network stability and bandwidth that it provides. Although the connections for each participant machine are the same as within the Summit MacroCog system, the connection method has been altered with the newer managed switch. Each station is wired into the new PowerConnect 2428 switch via a CAT 5E cable. This system is capable of speeds up to 1 Gbps, with a network latency between participant/experiment stations and servers of 50 nanoseconds. This system with the current hardware can support 24 simultaneous connections.

## Managed Switch

The PowerConnect 2428 is a 24-port web-managed switch. This device has a web-managed interface that allows for control of the system. Below in Table 1. Port Assignments. is the current configuration of the switch with each port labelled to which machine it is connected to:

***Table 1. Port Assignments.***

| Port Number | Computer | IP Address |
|---|---|---|
| 1 | Participant Station 1 | 172.16.0.21 |
| 2 | Participant Station 2 | 172.16.0.22 |
| 3 | Participant Station 3 | 172.16.0.23 |
| 4 | Participant Station 4 | 172.16.0.24 |
| 5 | Participant Station 5 | 172.16.0.25 |
| 6 | Participant Station 6 | 172.16.0.26 |
| 7 | | |
| 8 | | |
| 9 | Experimenter Left | 172.16.0.11 |
| 10 | Experimenter Right (Assigned) | 172.16.0.12 |
| 11 | | |
| 12 | | |
| 13 | Gateway (optional) | 172.16.0.1 |
| 14 | | |
| 15 | Bridge (optional) | |
| 16 | | |
| 17 | Phobos | 172.16.0.3 |
| 18 | Deimos | 172.16.0.4 |
| 19 | Phobos | 172.16.0.3 |
| 20 | Deimos | 172.16.0.4 |
| 21 | Phobos | 172.16.0.3 |
| 22 | Deimos | 172.16.0.4 |
| 23 | Phobos | 172.16.0.3 |
| 24 | Deimos | 172.16.0.4 |

Switch IP Address: 172.16.0.2
Gateway Address: 172.16.0.1 (Ceri1-PC)

The router receives internet connections through a gateway machine located in the Graduate Room in Suite 165. This machine, CERI1-PC, is a Windows 7 machine with two network interface cards. The computer is programmed to share its internet connection with the test-bed. The shared connection is masked as the computer's own IP address on the ASU network. Directly connecting the test-bed to the ASU network is not possible. The ASU network will try to manage the network, and it is imperative that it does not attempt this. The network is specifically configured for our parameters. If there are complications with the internet connection sharing, please see your system administrator.

### Accessing the Managed Switch

To access the managed switch you will need to be on a physically networked machine. Once there, follow these instructions:

1. Open the web browser to: http://172.16.0.2
2. Type in the following credentials:
    a. Login: admin
    b. Password: Qnt02Pyu**
3. From here you will see the main home screen. Two lists appear. The left side of the screen shows four main categories of System, Switch, Statistics/RMON, Quality of Service. The right side contains the component list of each of the four main categories. Only the active category's listing is available on the right side. We'll start with the System Category:
    a. System Category Components
        i. General
            1. Asset – This is where you can change the switch name, location, time, date, and view the service tag, serial number, and MAC address.
                a. Current name is Charon.
            2. Versions – Software version information
            3. Reset – Resets the device. DO NOT DO THIS UNLESS ABSOLUTELY SURE. It will take quite the feat to reset it. (Refer to the user manual on how to properly configure this device after a reset.)
            4. Secure Mode – This locks the switch down so that no access is allowed to logins. This is not used in DEXTAR.
        ii. IP Addressing
            1. IP Interface Parameters – this is where you can set the IP address of the machine. If this is reset you will need to locate the correct IP of the router before able to log in. You may need to use the terminal connection via serial cable if you cannot locate the IP address. Refer to the user manual on how to accomplish this.
            2. This is currently set to 172.16.0.2.
        iii. Diagnostics
            1. Integrated Cable Test – Tests cables. Useful during yearly cleanings and systems checks.
            2. Optical Transceiver Diagnostics – This option is not available for DEXTAR.
        iv. The following remaining features are not used for DEXTAR.
            1. Management Security
            2. SNMP
            3. File Management
            4. DHCP Server
            5. Advanced Settings (Jumbo Frames)

b. Switch Category Components
    i. Network Security – Network security is contained in the virtual environment and is not enabled through the router.
    ii. Ports
        1. Port Configuration – This allows you to configure each port. Each port is listed by the letter "g" then the port number associated on the device.
    iii. The following components are not enabled in DEXTAR:
        1. Address Tables
        2. Spanning Trees
        3. VLAN
        4. Link Aggregation
        5. Multicast Support

c. Statistics/RMON Category Components
    i. This section is for viewing statistics. DEXTAR monitors this internally through the servers as well. Refer to the manual for this section.

d. Quality of Service
    i. This category is completely disabled. It is not used for DEXTAR.

### Switch Configuration

The switch has been configured to be a basic switch with almost zero interference. The majority of network management was utilized through VMWare's ESXi and vCenter systems for this development. In larger developments, the VLAN system in the switch will likely need to be utilized more heavily than it is here.

## Servers

The DEXTAR servers both have four network connections to the switch. This is to allow the servers to better managed their bandwidth and input/output thresholds. This system is managed within the servers' vCenter Management System (see vCenter Management System). To control the management of this system:

1. Open vSphere client.
2. In the Navigation Bar, select: Home → Inventory → Hosts and Clusters.
3. Select the desired host
    a. Phobos – 172.16.0.3
    b. Deimos – 172.16.0.4
4. Click on the "Configuration" Tab.
5. In the left side bar menu, select "Networking".
6. From here you'll be able to access the properties menu for each of the virtual networks, hosts, and virtual machines.

Alternatively, you may change the IP address of the hosts from the management console on the ESXi host. To do this:

1. Attach a monitor and keyboard to desired host.
    a. Top server: Phobos
    b. Bottom server: Deimos
2. The system will be in a standby mode. Simply press any key and the monitor will brighten and display options.
3. You will need to log into the system. Press F2 to log in.
    a. Login: root
    b. Password: Qnt02Pyu**
4. Once in the menu, go to network configuration.
5. From this menu you will be able to change the IP of the server. The server will only take 1 IPv4 and 1 IPv6 address.
    a. For the purposes of DEXTAR IPv6 is not utilized.

## Non-Connected Computers

For DEXTAR it was not imperative or even practical to connect all of the machines to the DEXTAR network. Three machines were left off of the network for DEXTAR. These are:

1. The Aventura DVR
2. Experimenter Right
3. Summit Server

There is no practical gain by having these three machines constantly connected to the DEXTAR network. Instead, it leaves these machines open to damage due to the nature of the DEXTAR environment, despite best efforts. These machines are left connected to the original switch for the MacroCog test-bed. If there is need for these machines to interact with the DEXTAR network, please read the section on Bridged Connection.

## Bridged Connection

At times it may become necessary to connect the Summit Server up to the test-bed once again for demonstration or experimental purposes. There is an Ethernet cable in the experimenter left station. Attach this cable to any open port on the MacroCog switch, and to Port 15 on the DEXTAR managed switch.

Once attached, changes will need to be made to each computer on the extended network. The following IP addresses will need to be assigned:

1. Summit Server: 172.16.0.5
2. DVR: 172.16.0.31
3. Experimenter Right: 172.16.0.12

To change an address of a windows machine, do the following steps:

1. Open the Control Panel through the start menu (or other preferred method).
2. Go into Network Connections.
3. Right click on the connection you want to change.
4. A new window will pop up. You will need to select “TCP/IP Protocol”. It may have IPv4 listed, this is correct.

5. Change the IP address by selecting manual address, then enter the address. Here is the following configuration needed:
    a. IP Address: 172.16.0.XXX (Find above, nothing is assigned to addresses above 172.16.0.30, therefore you may use those liberally).
    b. Subnet Mask: 255.255.255.0 (or /24)
    c. Gateway: 172.16.0.1
    d. DNS: 129.219.17.200; 129.219.17.5; 129.219.13.81
    e. Search Domain: dhcpeast.asu.edu

It is not necessary to place a bridge cable if only one machine needs to be added to the DEXTAR network, such as the server. Instead, one may simply move the cable for the server from the MacroCog switch to the DEXTAR switch.

***IMPORTANT!*** If you connect the summit server, DVR, or Experimenter Right to the DEXTAR test-bed the following virtual machine TYPES must be off:

1. ***All Backtrack VMs***
2. All vyatta router appliances.

This is important to maintain the security of the network and the newly attached machines. To shut down these virtual machines, go into each machine and commence their respective shutdown sequence. These are as follows:

1. Backtrack
    a. Open a terminal window
    b. Command input: "poweroff"
2. Vyatta
    a. Command input: "sudo poweroff"
    b. If a password is required, use the respective password for each machine.

If this process is not possible, go into vSphere Client and right click on the VMs and proceed to power off. It is possible to write a command script to shut down these machines that can be implemented when necessary. See the section on writing a command script for more information.

## Virtual Networking

This development employs two types of networks, a physical standard network, and a virtual network. The virtual network is capable of being managed by the managed switch, however, for the current configuration vCenter will monitor all virtual networks.

This build consists of one virtual network with an IP address range that extends through all of the 10.X.X.X IP range. This virtual network, while managed by the physical servers, cannot access virtual machines not within this virtual network. Only machines assigned to a correct IP address within the virtual network may access it. Machines assigned to the virtual network do not have internet access. However, machines assigned to the physical network do have internet access.

***WARNING!*** It is possible to change IP addresses from a physical IP address to a virtual IP address and access either system. It is imperative that one does not accidentally due this as the consequences could be severe.

The virtual network for the DEXTAR scenario will be explained in detail within the section on the Test Scenario.

It should be noted that the DEXTAR network is entirely flat. This is for simplicity in programming and to reduce conflicts at this stage. Expansion of DEXTAR will require more network configuration reducing this from a flat network to a tiered network.

## Virtual Machines on Physical Network

Although this practice should be kept to a minimum, it was important to have two virtual machines present on the physical network. This is done by assigning a virtual machine an address otherwise reserved for a physically networked machine on the 172.16.0.X network. These two virtual machines are:

1. Administrative VM
    a. From this virtual machine, experimenters have access to the Nessus Scanner, Cauldron, and vSphere client from inside the test-bed. Other administrative functions should be added to this machine if needed.
    b. IP Address: 172.16.0.14
2. vCenter Server Appliance
    a. Although virtual, this server appliance runs the entire virtual environment and allows the servers to interact and be managed. Without this virtual machine, neither server would be functional.
    b. IP Address: 172.16.0.202
        i. This address was randomly assigned during reconstruction and could not be changed. It is not suggested that anyone changes this IP address as it will destroy the system certificates.

## Network Map

Below in Figure 3. Current Configuration of the DEXTAR Network with optional MacroCog Network Connection. a network map shows the physical network configuration including the two virtual machines attached.

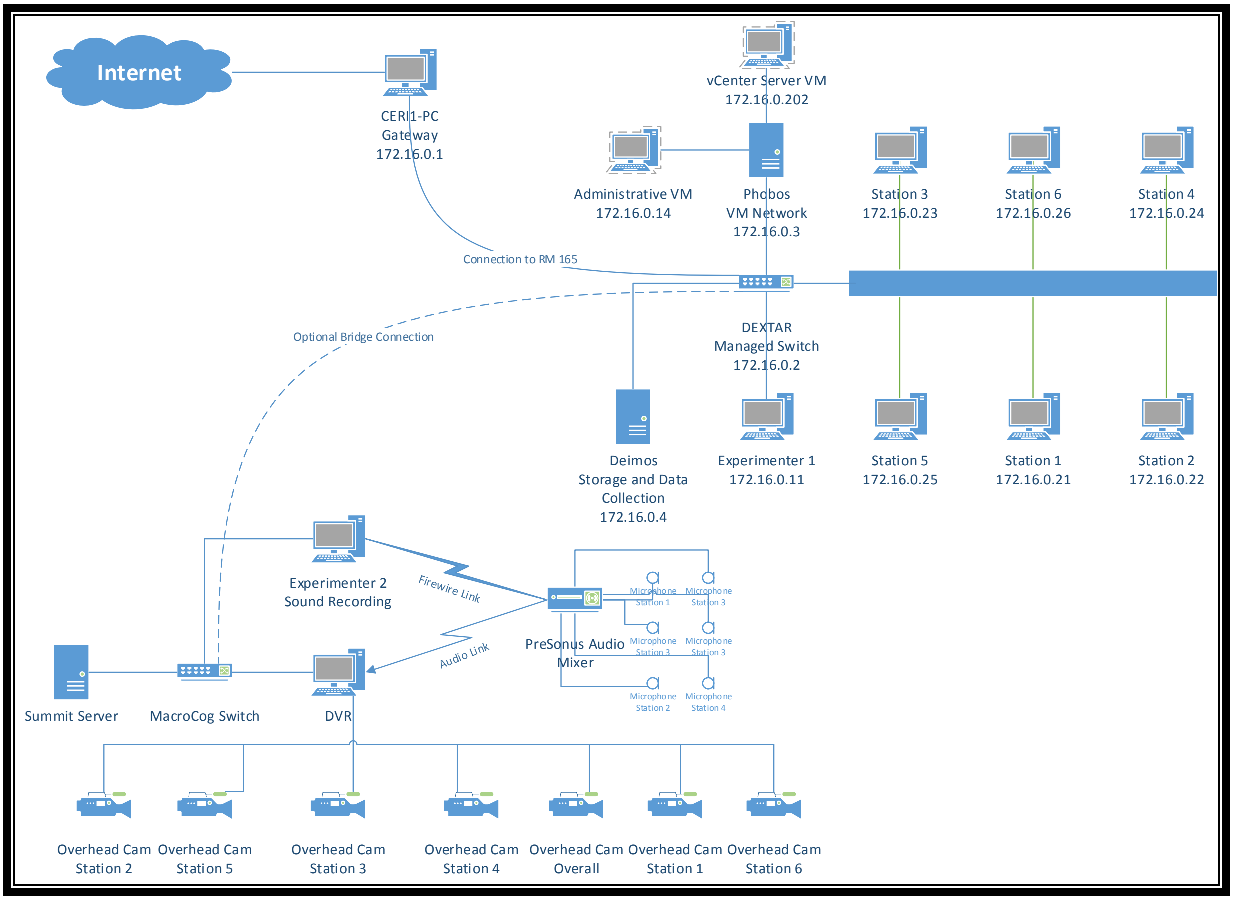


***Figure 3. Current Configuration of the DEXTAR Network with optional MacroCog Network Connection.***

# Software for Client Computers

Depending on the computer in question, certain software has been installed. This section will go through each computer and outline the software installations, changes, and current configurations.

## Participant Computers

### Operating System

The operation system of each of the six participant computers has been installed to Ubuntu 12.04 LTS 64-bit ("Precise Pangolin"). This software package is a long-term support package that receives continual updates from Ubuntu and the Linux community until April 2017. This OS for each machine is referred to as the "Host OS".
The download is available at: http://www.ubuntu.com/download/desktop A live CD may be made from this download and installed accordingly. This was the process used to develop these machines.

### Updates to OS

The updates to the OS come via the internet. These updates should be maintained as the host OS is not the subject of experimentation within this test-bed. It is vital that the security updates and program updates are maintained.

### Installation Notes

These machines contain an AMD 64-bit processor. Originally the 32-bit installation was selected, but was overwritten with the 64-bit installation to help with potential advantages this upgrade could provide for the test-bed.

During the OS installation, the following login credentials were used for each of the participant machines:
Login: Participant
Password: password
The OSes are programmed to automatically login. In addition to the participant login, and administrative login was also created with the following details:
Login: Admin
Password: D3xt4T3stb3d2013!

During installation, each machine was assigned to its preassigned IP address listed in the table in the section on the Managed Switch.

After installing this OS, the Logitech Bluetooth Keyboard and Mouse combination failed to be recognized correctly. This is due to a known error in programming the way that the OS receives the information. There is a solution that was implemented and all keyboard/mouse combinations work correctly. These are the steps to correcting this problem:

1. Open a terminal command window. This is done through the Ubuntu Menu in the upper left hand corner. Type in "Terminal".
2. Once open, type the command:

```
gksudo gedit /lib/udev/rules.d/97-bluetooth-hid2hci.rules
```

3. From here, find the line under “Logitech Devices” listed as a Kernel entry: ‘Kernel == “hiddev*”. Change this line to read ‘KERNEL == “hidraw”’.

Following the above fix, the Front USB ports were disabled via the BIOS setup. This is a security measure. To re-enable the front USB ports, simply restart the machine and press the F2 key during the BIOS splash screen to enter BIOS set up. Then under onboard devices, change the Front USB to “Enable”.

An additional change to the BIOS setup was the activation of “Wake On Lan” or WOL. This was accomplished by selecting network devices and allowing the wake on lan command function. This allows the computers to receive “Magic Packets” that instruct the computer to turn on. PLEASE NOTE, that this function only works with complete MAC addresses. Below is a table of MAC address (Table 2):

***Table 2. Table of MAC Addresses for Participant Computers***

| Station | MAC Address | IP Address |
|---|---|---|
| 1 | 00:1d:09:1d:6c:61 | 172.16.0.21 |
| 2 | 00:1d:09:1b:19:41 | 172.16.0.22 |
| 3 | 00:1d:09:1d:49:73 | 172.16.0.23 |
| 4 | 00:1d:09:16:e2:c7 | 172.16.0.24 |
| 5 | 00:1d:09:1b:b8:7c | 172.16.0.25 |
| 6 | 00:1d:09:17:99:48 | 172.16.0.26 |

## Software

Each of the six machines was designed to be identical. Although every effort was taken ensure that each machine was identical for the ease of use, there may undoubtedly be differences that were not intentional. Each machine has the following list of software:

1. Ubuntu 12.04 LTS 64-Bit OS
2. TeamSpeak 3
3. VMWare Player
4. Wireshark

## Software Installation

Each piece of software has a slightly different method of installation. The installs are explained here.

### VMWare Player

VMWare Player can be downloaded from:

http://www.vmware.com/products/player/

To install VMWare, a *.bundle file was downloaded. The following command was used to deploy the download installer package:

```
sudo sh VMware-Player-2.5.1-126130.i386.bundle
```

This function generated the installation package and installed the VMWare player.

### TeamSpeak 3

TeamSpeak 3 can be downloaded from:

http://www.teamspeak.com/?page=downloads

The Linux Client x86 3.0.10.1 version was selected for installation. To install, the following steps were taken:

1. The *.run install file must be changed into an executable allowable function:

```
sudo chmod +x TeamSpeak3-Client-linux_x86-3.0.10.1.run
```

2. The run file was then executed:

```
sudo ./ TeamSpeak3-Client-linux_x86-3.0.10.1.run
```

This initiated an unpacking program to unpack the files.

3. The unpacked files were then copied into the /opt/ file folder.

```
sudo cp –r TeamSpeak3-Client-linux_x86-3.0.10.1 /opt/
```

4. The files were then executed to ensure proper function:

```
sudo sh ts3client_runscript.sh
```

### Wireshark

Wireshark is a program designed to capture the network traffic of the network interface device. This is an important piece of software within the DEXTAR test-bed. The following steps were taken to install Wireshark:

1. Installing wireshark using this command: (This command initiates a download from the internet and an automated process to install the software. No further steps are required to install it.)

```
sudo apt-get install wireshark
```

2. At this point, wireshark can only run as an administrator. The following steps change this fact.
3. Create wireshark goup

```
sudo groupadd wireshark
```

4. Add username to the wireshark group (user_name= participant)

```
sudo usermod –a –G wireshark participant
```

5. Change the group ownership of file dumpcap to wireshark

```
sudo chgrp wireshark /usr/bin/dumpcap
```

6. Change the mode of the file dumpcap to allow execution by the group wireshark

```
sudo chmod 750 /usr/bin/dumpcap
```

7. Grant capabilities with setcap

```
sudo setcap cap_net_raw,cap_net_admin=eip /usr/bin/dumpcap
```

8. Verify.

```
sudo getcap /usr/bin/dumpcap
```

9. Logout and log back in.

Start-up Programming

The VMWare Player and TeamSpeak 3 programs were added into the start-up programs list. In the upper right hand corner, there is a settings icon with a drop down menu. In this menu "Startup Applications" was selected. From there, two commands to start the programs were added. The command for adding in VMWare Player was added into the startup applications settings using:

```
Vmplayer
```

The command for adding in TeamSpeak 3 was added into the startup applications settings using:

```
sh ts3client_runscript.sh
```

The command for adding in Wireshark is:

```
wireshark
```

Descriptions of each were added for ease of use. These applications now load once the computer is started/restarted.

## Experimenter Computer Left

The only experimenter computer changed for this test-bed as the experimenter left computer. The experimenter right computer relies on a firewire connection to record audio. While Linux can handle this type of connection, the recording program, CuBase, does not support a Linux OS. Therefore only the experimenter left computer was changed. This computer received an identical 1TB hard drive as those from the participant computers.

### Operating System

The operation system of each of the six participant computers has been installed to Ubuntu 12.04 LTS 64-bit ("Precise Pangolin"). This software package is a long-term support package that receives continual updates from Ubuntu and the Linux community until April 2017. This OS for each machine is referred to as the "Host OS".

The download is available at: http://www.ubuntu.com/download/desktop A live CD may be made from this download and installed accordingly. This was the process used to develop these machines.

## Updates to OS

The updates to the OS come via the internet. These updates should be maintained as the host OS is not the subject of experimentation within this test-bed. It is vital that the security updates and program updates are maintained.

## Installation Notes

These machines contain an AMD 64-bit processor. Originally the 32-bit installation was selected, but was overwritten with the 64-bit installation to help with potential advantages this upgrade could provide for the test-bed.

During the OS installation, the following login credentials were used for each of the participant machines:
Login: Experimenter
Password: password
The OSes are programmed to automatically login. In addition to the participant login, and administrative login was also created with the following details:
Login: Admin
Password: D3xt4T3stb3d2013!

During installation, each machine was assigned to its preassigned IP address listed in the table in the section on the Managed Switch.

After installing this OS, the Logitech Bluetooth Keyboard and Mouse combination failed to be recognized correctly. This is due to a known error in programming the way that the OS receives the information. There is a solution that was implemented and all keyboard/mouse combinations work correctly. These are the steps to correcting this problem:

4. Open a terminal command window. This is done through the Ubuntu Menu in the upper left hand corner. Type in “Terminal”.
5. Once open, type the command:

```
gksudo gedit /lib/udev/rules.d/97-bluetooth-hid2hci.rules
```

6. From here, find the line under “Logitech Devices” listed as a Kernel entry: ‘Kernel == “hiddev*”. Change this line to read ‘KERNEL == “hidraw”’.

Following the above fix, the Front USB ports were disabled via the BIOS setup. This is a security measure. To re-enable the front USB ports, simply restart the machine and press the F2 key during the BIOS splash screen to enter BIOS set up. Then under onboard devices, change the Front USB to “Enable”.

## Software

Each of the six machines was designed to be identical. Although every effort was taken ensure that each machine was identical for the ease of use, there may undoubtedly be differences that were not intentional. Each machine has the following list of software:

5. Ubuntu 12.04 LTS 64-Bit OS
6. TeamSpeak 3
7. VMWare Player

8. Wireshark

## Software Installation

Each piece of software has a slightly different method of installation. The installs are explained here.

### VMWare Player

VMWare Player can be downloaded from:

http://www.vmware.com/products/player/

To install VMWare, a *.bundle file was downloaded. The following command was used to deploy the download installer package:

```
sudo sh VMware-Player-2.5.1-126130.i386.bundle
```

This function generated the installation package and installed the VMWare player.

### TeamSpeak 3

TeamSpeak 3 can be downloaded from:

http://www.teamspeak.com/?page=downloads

The Linux Client x86 3.0.10.1 version was selected for installation. To install, the following steps were taken:

1. The *.run install file must be changed into an executable allowable function:

```
sudo chmod +x TeamSpeak3-Client-linux_x86-3.0.10.1.run
```

2. The run file was then executed:

```
sudo ./ TeamSpeak3-Client-linux_x86-3.0.10.1.run
```

This initiated an unpacking program to unpack the files.

3. The unpacked files were then copied into the /opt/ file folder.

```
sudo cp -r TeamSpeak3-Client-linux_x86-3.0.10.1 /opt/
```

4. The files were then executed to ensure proper function:

```
sudo sh ts3client_runscript.sh
```

### Wireshark

Wireshark is a program designed to capture the network traffic of the network interface device. This is an important piece of software within the DEXTAR test-bed. The following steps were taken to install Wireshark:

1. Installing wireshark using this command: (This command initiates a download from the internet and an automated process to install the software. No further steps are required to install it.)

```
sudo apt-get install wireshark
```

2. At this point, wireshark can only run as an administrator. The following steps change this fact.
3. Create wireshark goup

```
sudo groupadd wireshark
```

4. Add username to the wireshark group (user_name= participant)

```
sudo usermod –a –G wireshark participant
```

5. Change the group ownership of file dumpcap to wireshark

```
sudo chgrp wireshark /usr/bin/dumpcap
```

6. Change the mode of the file dumpcap to allow execution by the group wireshark

```
sudo chmod 750 /usr/bin/dumpcap
```

7. Grant capabilities with setcap

```
sudo setcap cap_net_raw,cap_net_admin=eip /usr/bin/dumpcap
```

8. Verify.

```
sudo getcap /usr/bin/dumpcap
```

9. Logout and log back in.

### Start-up Programming

The VMWare Player and TeamSpeak 3 programs were added into the start-up programs list. In the upper right hand corner, there is a settings icon with a drop down menu. In this menu "Startup Applications" was selected. From there, two commands to start the programs were added. The command for adding in VMWare Player was added into the startup applications settings using:

```
Vmplayer
```

The command for adding in TeamSpeak 3 was added into the startup applications settings using:

```
sh ts3client_runscript.sh
```

The command for adding in Wireshark is:

```
wireshark
```

Descriptions of each were added for ease of use. These applications now load once the computer is started/restarted.

## Customized Scripts

Turn On Test-Bed

The experimenter left computer contains two Linux Bash scripts that allow for the control of the six participant computers. The first script turns on all computers within the test-bed through the use of “magic packets”. See Table 2 for a list of MAC address for participant machines. This script has two locations within Experimenter Left. The first location is on the Desktop in a folder called ‘Scripts’. This script is run in a terminal by the following command:

```
sudo sh TurnOn
```

The script is as follows:

```
#!/bin/bash
# 1
wakeonlan 00:1d:09:1d:6c:61
# 2
wakeonlan 00:1d:09:1b:19:41
# 3
wakeonlan 00:1d:09:1d:49:73
# 4
wakeonlan 00:1d:09:16:e2:c7
# 5
wakeonlan 00:1d:09:1b:b8:7c
# 6
wakeonlan 00:1d:09:17:99:48
#Statement
notify-send "All Stations on."
exit
```

This script turns on all participant machines with the “wakeonlan” terminal command. The previous command initiates the script as an administrator. Once the command has been issued, the computer will ask for the password. This is previously stated above.

The second location of the TurnOn (and TurnOff, mentioned later) script is in the **/usr/bin/** location of the computer. This location is set aside for applications and scripts. This script has been transformed into an application that can be launched from the side bar by clicking on the **Sun** icon. This starts the automated process of remotely starting every machine which is similar to the manual process. This script is not different within the function or programming. However, the icon is a small application that calls the script to function. This is done with a separate script that is located in **/usr/bin/** as well as the TurnOn shell script.

```
[Desktop Entry]
Type=Application
Name=TurnOn
Exec=/usr/bin/TurnOn
Icon=/usr/share/icons/Humanity-Dark/status/48/weather-clear.svg
```

It is possible, as with the TurnOff script below, to implement these scripts remotely through remote connections into the system. More on this in the Remote Connections section.

Turn Off Test-Bed

The second script turns off the test-bed and is titled ‘TurnOff’. This script turns off all of the participant computers in the test-bed. This script has two version as with the previous script. The first version located in the Desktop folder ‘Scripts’ can be run in the same fashion as the TurnOn script. Simply use this command from a terminal window:

```
 sudo sh TurnOff
```

The script is as follows:

```
#!/bin/bash
# Statement
notify-send "Shutting down participant stations."
# 1
expect -c 'spawn ssh -t participant@172.16.0.21 sudo poweroff;expect
password;send "pass$
# 2
expect -c 'spawn ssh -t participant@172.16.0.22 sudo poweroff;expect
password;send "pass$
# 3
expect -c 'spawn ssh -t participant@172.16.0.23 sudo poweroff;expect
password;send "pass$
# 4
expect -c 'spawn ssh -t participant@172.16.0.24 sudo poweroff;expect
password;send "pass$
# 5expect -c 'spawn ssh -t participant@172.16.0.25 sudo poweroff;expect
password;send "pass$
# 6
expect -c 'spawn ssh -t participant@172.16.0.26 sudo poweroff;expect
password;send "pass$
# Statement
notify-send "Test-bed Shutdown"
exit
```

The second script, which is more automated,  is deployed by pressing the **Moon** icon on the side bar. This script is slightly different from before and adds in the **gnome-terminal x** command which informs the computer to initiate this command as if it was in a command terminal window, without having to open one. This script shuts down the entire participant test-bed, but does this in one of two ways. You may notice no indication of performance other than the message “Test-bed Shutdown”, or you may notice a series of windows opening and then closing very fast. Both behaviours are normal and are equal in performance. The second script is as follows:

```
 #!/bin/bash
# Statement
notify-send "Shutting down participant stations."
# 1
gnome-terminal -x expect -c 'spawn ssh -t participant@172.16.0.21 sudo
poweroff;expect p$
# 2
gnome-terminal -x expect -c 'spawn ssh -t participant@172.16.0.22 sudo
poweroff;expect p$
# 3
```

```
gnome-terminal -x expect -c 'spawn ssh -t participant@172.16.0.23 sudo
poweroff;expect p$
# 4
gnome-terminal -x expect -c 'spawn ssh -t participant@172.16.0.24 sudo
poweroff;expect p$
# 5
gnome-terminal -x expect -c 'spawn ssh -t participant@172.16.0.25 sudo
poweroff;expect p$
# 6
gnome-terminal -x expect -c 'spawn ssh -t participant@172.16.0.26 sudo
poweroff;expect p$
# Statement
notify-send "Test-bed Shutdown"
exit
```

As with the TurnOn script, a separate script is used to call this script to function through the icon. This script is located in **/usr/bin/** as well with the TurnOff script. The script is as follows:

```
[Desktop Entry]
Typ=Application
Name=TurnOff
Exec=/usr/bin/TurnOff
Icon=/usr/share/icons/Humanity-Dark/status/48/weather-clear-night.svg
```

Both of these scripts have Icons listed within their configurations and were placed onto the sidebar for easy usage.

# DEXTAR Servers Configuration

The DEXTAR system functions off of two paired ESXi servers. Within the software environment these two machines are paired into a Data Center, called DEXTAR, and Cluster, aptly named Mars. The entire operating environment is provided by VMware and can be downloaded through the Sandia Research Corporation log in information:

- Website: https://my.vmware.com/web/vmware/login
- Login: slambert@sandiaresearch.com
- Password: OSDCyber2013

## Operating System

The Operating Software of DEXTAR is VMWare vSphere 5.1. The servers are installed with the Operating system called ESXi. The installation of the ESXi system into the servers was simple. The servers were delivered to ASU without any preinstalled software other than the RAID Array[2]. The ESXi was installed using the default configurations from the installation media. The only differentiation is that the top server was designated as Phobos and the bottom server was designated as Deimos. Their differences will be highlighted throughout the remainder of this manual. A short description follows:

- Phobos – This is the main runtime environment system for the DEXTAR Test Scenario.
- Deimos – This is the main runtime environment system for the DEXTAR management, data capture, and storage systems. This is also where the ISO repository is located.

After installation, basic configuration is possible directly at the host machines. To do this you will need to do the following:

1. Hook up a monitor and keyboard to desired host.
2. Press F2 to wake the machine and login.
   a. Login: root
   b. Password: Qnt02Pyu**
3. You will see a menu with a variety of options.
   a. Configure Password – You can configure the access passwords to the host from this menu.
   b. Configure Lockdown Mode – Don't touch this.
   c. Configure Management Network – This is how to configure the network. The management network is the title for the main network.
   d. Restart management network – Will likely not be required, but you may restart the network device from here. It will disconnect every connected physical and virtual machine.
   e. Test management Network – Tests the network.

[2] The RAID Array is a hardware controlled RAID 1. This is made up of two identical 2TB 7200rpm SAS hard drives. The RAID array controls are accessible from the BIOS splash screen. However, these are not to be changed or altered without prior knowledge to know how. Doing so will result in a catastrophic system failure. No further notes on the array will be mentioned.

f. Restore Network Settings – This will restore them to a saved configuration or default configuration
g. Restore Standard Switch – Only needed if the main virtual switch (vSwitch0) has been changed.
h. Configure Keyboard – Not likely needed. Used if the keyboard type is changed.
i. Troubleshooting Options – This item lets you change specific functions that *allow* for remote troubleshooting. There are no troubleshooting tools here.
j. View System Logs/Support Information – View the respective documents and logs.
k. Reset System Configuration – This will reset the system. Do not use this.

After the ESXi system has been configured the vSphere Client software that is used to access the servers and control the system can be downloaded either from VMware through the repository or directly from the ESXi servers on their respective IP addresses (http://172.16.0.3). Download and install the vSphere Client.

At times it may become necessary to access a host directly through vSphere rather than through a management console. To access a host directly through vSphere:

1. Start vSphere.
2. Input the IP address for the desired host.
   a. Phobos – 172.16.0.3
   b. Deimos – 172.16.0.4
3. Input the login information:
   a. Login: root
   b. Password: Qnt02Pyu** (Possibly: D3xt4rT3stb3d2013!)
4. Press enter.
5. This will log you into the host directly. You will not have a multitude of options as if you went through a management console. Instead you can only directly control the virtual machines on this host, and only control this host. It is not possible to interact with the other host in this situation.

Once you have access to a host through vSphere client, you will be able to add virtual machines, configure the host and manage operations. However, the vCenter Management System is set up to run the management of the system.

## vCenter Management System

The centre management point of the DEXTAR system is the vCenter Management Server Appliance. This appliance is fully configured and can be downloaded from the aforementioned VMware repository. In order to install this appliance you will need to complete the vSphere download and log directly into the host that will receive the vCenter appliance. Currently, the vCenter appliance is loaded into, and operating from, Phobos at IPv4 172.16.0.202/24.

To load a vCenter Appliance:

1. Log into the desired host directly through vSphere.
2. Once logged in, in vSphere go to: File → Deploy OVF Template.
   a. Locate the vCenter Appliance.

    b. Open the appliance. This will load it into the server.

3. During this process an IP address will automatically be assigned. You may change this later at your own discretion.
4. The device will need to be powered on. To do this:
    a. Locate the device in the list of virtual machines within the host on the left hand navigation bar.
    b. Right click on the appliance, and a menu drops down.
    c. Click on "Power On".
        i. Alternatively, you can select the appliance and press **CTRL + B.**
5. Once installed and powered on, accessing the device directly through the console will only point you to the web address. From a networked machine, go to the web address: https://X.X.X.X:5480. Where the X.X.X.X is the current IP address of the appliance.
    a. If you need to determine the IP address, once the appliance is selected, click on the **Summary** tab in the right portion of the screen (the main section).
    b. The IP address will be listed under "IP Addresses".
6. To access the website, the following login details are required.
    a. Login: Root
    b. Password: vmware
7. In the DEXTAR test-bed, the vCenter password was changed immediately to: D3xt4rT3stb3d!

Once the vCenter appliance was loaded and operational, the servers were added into the system. This process was completed by:

1. There are two processes to adding servers to the vCenter appliance. This first method is through the web client.
    a. The client can be accessed from: https://172.16.0.202:9443
    b. The login is the same as above
        i. Login: root
        ii. Password: D3xt4rT3stb3d!
    c. From here, you have the same functionality as with the desktop client. However, only instructions for the desktop client will be provided.
2. The second method is the vSphere Desktop Client. Once the vCenter Appliance is operational, log out of the host system. You will need to log into the vCenter appliance directly. You will do this from now on for any function with DEXTAR.
    a. Open your installed vSphere Desktop Client
    b. Login Address: 172.16.0.202
    c. Login ID: root
    d. Login Password: D3xt4rT3stb3d!
    e. This will load the inventory and functionality of vCenter.
    f. Please note that the first login often takes some time compared to subsequent logins.
3. Once fully loaded, select the **Home → Inventory → Hosts and Clusters** navigation section from the top navigation bar.

4. Select the vCenter Appliance in the left navigation tree. In DEXTAR, this appliance is named Eris.
5. Select **Create a Datacenter**.
    a. Follow onscreen instructions and customize accordingly. For this installation, default settings were used, and the Datacenter was named DEXTAR.
6. Once the Datacenter is created, click on **Create a Cluster.** You will need to create a cluster to assign the servers to.
    a. Follow onscreen instructions and customize according. For this installation, default settings were used, and the Cluster was named Mars.
    b. It should be noted that this license of VMware does not contain vSphere HA or vSphere DRS.
7. Once the Cluster is created, click on **Add a Host.**
    a. You will need the IP address of the host. For instance:
        i. Host IP (Phobos): 172.16.0.3
    b. You will need authorization information. This is the root login:
        i. Login: root
        ii. Password: D3xt4rT3stb3d!
    c. Once these are inputted, click OK.
    d. A summary will be provided of the host's details.
    e. Click ok and finish adding the host. You may add as many hosts as you would like.
        i. Please note that the license only works for three physical machines with 2 physical CPUs each.

Once those operations are completed you will have successfully added both hosts to the vCenter appliance. This appliance does not currently administer load balancing. This was a design decision and not a system limitation.

vCenter is the central management system for the VMware software. You will need to directly access vCenter when working with the system for administrative and design purposes. During scenarios participants are NOT to have access to vCenter. This creates a very real security hole. No passwords similar or mimicking the vCenter and Host passwords should be used throughout the Test Scenarios.

## Data Storage

Within DEXTAR there are several differentiations between data storage sections. For program installation media, a data store was used located on Deimos (172.16.0.4). This was to free up hard drive space from Phobos which holds the majority of the test scenario and VMs.
To access the ISO (installation media) datastore:

1. Log into the vCenter appliance through vSphere desktop client.
2. Navigate using the top navigation bar to **Home → Inventory → Datastores and Datastore Clusters.**
3. You may select either host to see the datastore.

    a. Please note, both servers have datastores by default. The Deimos datastore was used for installation media, although Phobos does contain a few installation media. This is left from the initial configuration and installation of VMs to increase speed of installation.
4. Once you've selected a datastore from the left navigation tree, the first tab on the right screen **Getting Started** will have a link entitled **Browse Datastore**. Click this link.
5. From here a secondary window opens up to show a navigation through the host.
    a. Each folder listed is either a VM or a folder that contains further information.
    b. For the ISO storage you will find the ISOs listed in folder *ISO Storage*.
6. Click on **ISO Storage.** This will bring up a list of available ISOs within this storage folder.

You may place any file into the datastore. **DO NOT** place any virus definitions, encyclopaedias, or reference files. To place files into the datastore:

1. Go into browsing the datastore you wish to upload the file.
2. Select, or create, the folder you wish to upload to and enter the folder.
3. You will need to click on the upload icon (an image of a storage disc with a green arrow pointing upwards):
4. Once clicked, a smaller menu will ask File or Folder. Once you've selected your desired upload type, you will then need to locate the file or folder to upload.

It is also possible to download from the datastore. To download:

1. Go into browsing the datastore you wish to download from.
2. Select the folder, and the file you wish to download.
3. You will need to click the download icon (an image of a storage disc with a green arrow pointing downwards):

You will need to upload ISOs for installation media in order to use them with virtual machines. Please note that only ISO and ISO-type files are allowed to be used as installation media. This is due to ESXi using the ISO as a virtual CD and is only programmed to allow that or similar file types for virtual CD mounting. This applies mostly to Operating Systems. Below in Table 3 is a list of currently available OSes within the Deimos datastore:

***Table 3. Available ISO OS Software***

| Available ISO OS Software |
|---|
| Windows 7 Ultimate Edition |
| Backtrack 5 R3 |
| CentOS |
| WinXP Pro Service Pack 0 (Floppy Image, not ISO) |
| Vyatta Switch/Router |
| Ubuntu 12.04 LTS |
| "Damn Vulnerable Linux" |
| Windows Server 2008 |
| Windows Server 2003 R2 |

# VMware Environment

The VMware environment is the main environment that the DEXTAR test bed runs off of. This environment controls nearly all of the virtual machines deployed within this installation. At the time of writing, there were nearly 100 different virtual machines, templates, and base installations enabled within the DEXTAR system. As such, it is important to provide an overview of the VMware environment.

This portion of the manual is broken down into a few sections:

1. Virtual Machine Installation
2. Virtual Machine Customization
    a. Hardware
    b. Adding Hardware
    c. VMware Tools
    d. Additional Hardware
3. Virtual Network Structure

## Virtual Machine Installation

Before proceeding, the following guide will instruct a user in how to install a virtual machine. Because of the length of the instruction, only adaptations to the procedure will be specifically mentioned in subsequent sections. If no adaptations are mentioned, this guide is to be followed to install the desired virtual machine.

1. Open vSphere client on the main Gateway console, and log into the vCenter Server.
2. On the left side navigation tree, select the **DEXTAR** datacentre.
3. Go to **File → New → Virtual Machine.** (Alternatively, you may press **CTRL + N**).
4. A new window will open up to set up the virtual machine.
5. Select **Typical** settings and press next.
6. Next, select the location of where to place the Virtual Machine. You may use folders and groups to help form a structure within vSphere to be visually more categorized.
    a. To make a folder, you'll need to exit the installation of the virtual machine.
    b. Right click on the **DEXTAR** datacentre and a drop down menu will appear.
    c. Select **New Folder**.
    d. Then rename the folder and move as desired to create a structure.
7. Next, select **Mars** as the cluster for installation.
8. Next, select which host you wish to install the VM on.
    a. For test-scenario VMs, install on Phobos.
    b. For test-monitoring, administrative, and template design, install on Deimos. #
    c. **Please Note** you must install the virtual machine onto the host where the ISO is stored that you want to install. The datastores are not available between the hosts.
9. The next window will indicate the storage device to be used. There is only one storage device per server, so only one option should be available. Click next.

10. From here, select the type of operating system the virtual machine will use. It is important to be as accurate as possible as this creates the correct program environment.
    a. For Vyatta and Damn Vulnerable Linux Installations, select **Other Linux (32-bit).**
    b. For other installations, find the operating system from the drop down.
11. Next you will be able to assign the number of network cards you will need.
    a. Most installations will only require one.
    b. Vyatta installations will require between 2 and 4 each. This will depend on the router's intended use and design practices.
    c. Some server installations may require 2. In the case of the Active Directory Server in this Test Scenario, two adapters were used.
    d. You may add more adapters later if you are unsure of the desired amount. You may also delete adapters that are not needed.
12. The next step is to create the virtual disk. This is an allocated space on the hard drive of the host machine. For all installations unless otherwise stated, use "Thin Provision".
    a. Thin provisioning is a method of reducing allocated hard drive space from the host to virtual machines. This process sets a virtual machine to believe it has X GB of hard drive space but will only physically use as much hard drive space as physically required by the programming. The disk then expands to match the required needs of the virtual machine as it goes. This allows for the allocation of more storage space than physically present.
13. The final step is finalization. In this step you may review your settings. You are at this point able to return to the previous screens/options to change anything desired.
14. It is also possible to further edit the virtual machine preferences by checking a small box below the information text box. This will take you to the customization screen of virtual machines. This will be discussed in more detail in the next guide.
15. Once customization is completed, if utilized, the system will create the virtual machine disk. This has not installed the OS though. To do this, you will need to go into the settings of the disk and allow for the ISO.
16. Right click on the newly created virtual machine from the left navigation window.
17. Click on **Edit Settings**.
18. This will open a new window where you will see all the components of the virtual machine. Click on **CD/DVD drive**.
    a. If this does not exist, you will need to add it. See Virtual Machine Customization in the Client computers section.
19. On the right hand side of the forefront window you will see a **Datastore ISO File** option. Select this radio button.
20. Click **Browse** to find your desired ISO file.
    a. **Important.** The ISO must be located on the same server as the empty virtual machine. You may move the virtual machine after installation to the other server.

Often you will create machines on Deimos and move the new virtual machine to Phobos.

21. Just above the Datastore ISO File selection, the device connection status box indicates if the component is functional and connected. Check both boxes in this area to allow the ISO file to function.
22. Once all selections have been made, click **OK.**
23. Power on the virtual machine.
    a. You may right click then select **Power → Power On.**
    b. Or you may simply select the virtual machine and press **CTRL + B**.

Once completed, the virtual machine will be constructed, the ISO loaded and ready for installation, and the virtual machine should be on. If those three aspects are not met, revisit these steps to ensure each of the three aspects are met before proceeding.

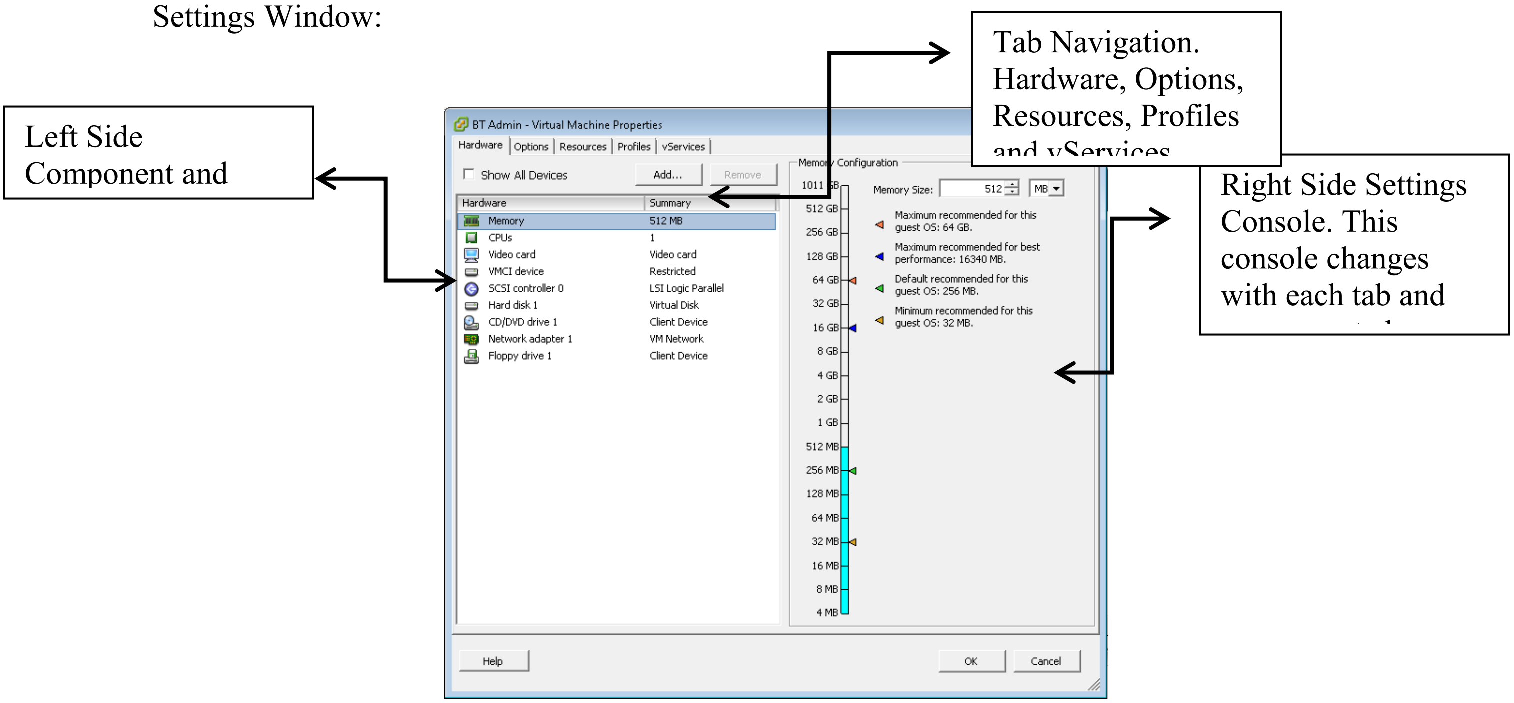
## Virtual Machine Customization

Once a virtual machine is installed, customization is often need to ensure the machine works as intended. This also allows you to add and delete virtual components of the system such as CD/DVD drives, USB devices, Network Adapters, Hard drives, and so on.

To access settings:

1. Select the intended virtual machine and right click.
2. Select **Edit Settings.**
3. This will open up a new window with all available settings.
    a. When a VM is powered on, fewer settings are able to be changed.
    b. When a VM is powered off, all settings are able to be changed.

Settings Window:



Tab Navigation. Hardware, Options, Resources, Profiles and vServices

Left Side Component and

Right Side Settings Console. This console changes with each tab and

BT Admin - Virtual Machine Properties

□ Show All Devices

| Hardware | Summary |
|---|---|
| Memory | 512 MB |
| CPUs | 1 |
| Video card | Video card |
| VMCI device | Restricted |
| SCSI controller 0 | LSI Logic Parallel |
| Hard disk 1 | Virtual Disk |
| CD/DVD drive 1 | Client Device |
| Network adapter 1 | VM Network |
| Floppy drive 1 | Client Device |

Memory Configuration

Memory Size: 512 MB

Maximum recommended for this guest OS: 64 GB.

Maximum recommended for best performance: 16340 MB.

Default recommended for this guest OS: 256 MB.

Minimum recommended for this guest OS: 32 MB.

***Figure 4. Virtual Machine Settings Window***

## Hardware

The first settings tab is **Hardware**. In this section you can add, remove, and change the state of virtual hardware. There are some limitations to this.

1. Windows machines cannot have their processor settings changed once the operating system is installed. Windows is incapable of handling this change.
2. Assigning resources to a virtual machine below its minimum spec requirements will result in a poor performance or hard crash.
3. Assigning more resources than the host has available, or running more virtual machines with resource configurations surpassing the physical specs of the hosts will result a host system failure.
    a. **Please note.** Avoid host system failure. Crashes of this magnitude are often damaging both physically as the system tries to compensate and within programming.
4. **Phobos** has more available Memory to allocate (48 GB) compared to **Deimos** (16GB). Phobos is the main runtime environment system.

Changing the virtual hardware of a virtual machine, within the confines of system limitations, is a simple process. The best time to change system configurations is during the initial setup. During installation/creation of a new virtual machine, the last step allows you to edit the hardware configuration of a virtual machine before creation. For example, let's change the memory and processor information while adding an additional network adapter.

1. The first step will be to adjust the memory. This is also the first setting available when opening up settings.
    a. **Please note.** The virtual machine must be off to change the processor and memory settings.
2. The Left Side Navigation screen should have **Memory** selected. The right hand settings console will have a bar with memory amounts along the left side.
3. The gauge will have four markers:
    a. Minimum Recommended for guest OS
    b. Default Recommended for guest OS#
        i. This option or above but below the next option are recommended and used within DEXTAR.
        ii. **Please Note.** You should not allocate more memory across all virtual machines than the host machine has available.
    c. Maximum Recommended for best performance for guest OS
    d. Maximum Recommended for guest OS.
4. There are two options for adjusting memory allocation.
    a. You may use the slider bar to click and drag the teal bar up and down the gauge to select the desired amount.
    b. Alternatively, and more precise, you may type the memory allotment into the text box that indicates the memory size allotted.

5. If the memory was the only aspect to configure, simply hit "OK" at the bottom of the pop-up window and the settings will be applied. If you have other settings to complete, such as changing the processor[3], you may simply select the next option.
6. In the Hardware list on the left side, select **CPUs** and look to the right side panel.
7. There are two options within the right side panel.
    a. Number of Virtual Sockets
        i. This is an indication to the VM how many physical CPU chips are installed. One socket is standard for desktops, laptops, and general VM use. Two or more sockets are reserved for server VM installations.
    b. Number of Cores per Socket
        i. This is the number of cores each virtual chip will contain. This selection may be varied dependent on the processing requirements of the machine. For example, Windows XP can run effectively on one or more cores, while Windows 7 runs more effectively on at least 4 cores.
8. Once you have completed your customization, click **OK** in the lower right hand corner.

## Adding Hardware

It is possible, and often required, to add additional hardware components to the virtual machine. The majority of virtual hardware can be added with relative ease.

1. To add hardware, open up the **Edit Settings** window to the VM you wish to add hardware to.
2. Once open, above the left-hand panel there will be two buttons for **Add** and **Remove.**
3. Click on **Add.**
4. From here you can add the following types of devices:
    a. Serial Port
    b. Parallel Port
    c. Floppy Disk
    d. CD/DVD Drive
    e. USB Controller
    f. Ethernet Adapter
    g. Hard Disk
    h. SCSI Device
5. Each of these choices has its own configuration settings. In general, setup the device for the functionality desired.
6. Once the device is configured, it will be added to the virtual machine after you click **OK** in the lower right hand corner.

[3] The processor must be changed before the installation of any Windows OS. This is not a factor for Linux OSes. As of the writing of this manual, it is not possible to change the processor in a Windows OS VM without reinstalling the OS itself.

7. **If an OS is NOT installed:** During installation the OS will find the hardware and install accordingly. If the device is not installed, it may be required to install the VMTools before the device can be utilized. This will be discussed in the following section.
8. **If an OS IS installed:** If the OS is already installed, it may recognize the new hardware during power-on and either prompt the user to install the driver, install the driver itself, or query the user for more information.

## VMware Tools

VMware vSphere comes with an add-on set of tools that may be installed into each VM. It is not required to install the VM Tools into each virtual machine. Installation does increase some of the interaction capabilities as well as certain VMware services. In particular, this allows vSphere to read certain information about the VM without the user needing to activate or go into the VM itself. To install VMware Tools:

**Please note**: The majority of DEXTAR does NOT have VM Tools installed.

1. It is possible to upgrade/install the VMware tools while a virtual machine is powered on. The virtual machine will need to be on in order to complete an installation of VM Tools.
2. From vSphere, right click on the virtual machine you would like to install the tools into.
   a. Go to **Guest → Install/Upgrade VMware Tools**.
3. This will load a virtual CD into the virtual machine.
4. If the VM is powered off:
   a. Power on the VM.
5. Go into the VM. It should have recognized that a disc was “inserted” into its drives.
6. Click **Run Installation.**
7. Follow the onscreen prompts.
8. The VMware Tools are now installed.

## Additional Hardware

Within virtual machines a variety of hardware is required, but not often shown. To view all hardware you will need to go into the settings of the virtual machine and click **Show All Hardware**. This will allow you to see all hardware including mouse and keyboard information. Not all hardware options are utilized for this installation. For example, **VMCI device** is a device for VMs to interact with one another as well as the server. This is not utilized due to potential security loopholes that may be present. Here is a list of additional hardware components:

1. Video Card
   a. The virtual machine’s video card. Options allow the number of displays and graphics memory to be changed. It is also possible to have the VM detect the number of monitors a computer system accessing the VM is utilizing.
2. VMCI
   a. This device allows a control throughput for the VMware environment to control the VM. There is optionality for this device to allow VMs to directly interact with one another.
3. SCSI Controller

a. This allows the sharing of harddrives among the virtual machines. It is possible to set this to no sharing, local server sharing, or global server sharing (any server connected to the data cluster is considered a part of global).

4. Hard Disk
   a. This is the VM’s hard disk, or rather virtual hard disk.
5. Floppy Drive
   a. A legacy device that emulates a 3.25” floppy drive.
      i. There is next to no purpose for this device.

## VMware Tools

VMware and the vSphere environment utilizes a selection of tools that are installed into the virtual machine called VMware Tools. These tools allow for greater interactions between the VMware environment and the virtual machine. This also allows for scripts, commands, and higher functions to occur from outside the virtual machine that can modify the machine. This implementation relies on the VMware tools to be installed for the functionality of this environment. Install these tools on every new virtual machine. To install:

1. Select the recently installed virtual machine and right click.
   a. Go to **Guest → Install/Upgrade VMware Tools**.
2. This will load a virtual CD into the virtual machine.
   a. If auto-run is setup within the virtual machine, it will likely begin to install the tools.
   b. If auto-run is not setup, you will need to navigate to the virtual CD within the OS, search the disc for **setup.exe** or **setup64.exe** if you are running a 64-bit virtual machine and operating system.
3. Follow on-screen instructions for installation.
   a. DEXTAR utilized the **Typical** setting for installation.
4. When prompted, click **Install**.
5. When complete, restart the virtual machine.
6. VMware tools are now active.

From time to time, it may be necessary to upgrade these tools. You can follow the steps above to upgrade the VMware tools, provided that the new set of tools has been downloaded off of the internet. If the servers have internet access, they may automatically download the new tools. If the servers do not, you will need to log into your VMware account and download the updates manually through your products download links.

## Virtual Network Structure

Within the VMware environment, it is possible to create multiple and complex network designs. However, this installation uses a “flat” network design. This means that all virtual machines and physical machines are connected to the same virtual network. This does two things:

1. Allows for complete interaction of virtual machines with physical machines if activated appropriately.
2. Provides for a simple network assignment within vCenter.

Although this set up has positive outcomes, there are a few drawbacks:

1. If improperly assigned, the test network may cause damage to the ESXi hosts.
2. If improperly assigned, the test network may be exposed to the Internet.

The main reason for this choice given the potential outcomes for this installation is due to licensing constraints. It is not possible within the implementation of VMware that this system utilizes to use virtual distributed switches, or allow for complicated networks to be built.

### Assigning the Virtual Network

Virtual machines created within this system will automatically be assigned to the main "management network" once provided with a Ethernet device. However, if this is not done properly this is how to manually set the virtual network:

1. Select the desired virtual machine in the left navigation panel of vSphere.
2. Right click on the VM and open **Edit Settings**.
3. In the right hand panel, there will be an option for **Network**.
4. In this drop down menu, all available networks will be listed.
5. Currently, this system utilizes **VM Network**. As such, this should be the selection unless a new network was created.

Since this installation did not create a functioning new network, this manual will forgo explaining this option. Please refer to the VMware vSphere 5.1 Instruction Guide for ways on developing more networks.

## Virtual Machine Template Creation

For DEXTAR, Virtual Machine templates were utilized to facilitate the creation of the test network, and for a measure to ensure similar operating system implementations. A template is a virtual machine blueprint, and can be made from any existing virtual machine. For DEXTAR, a number of templates exist (see Table 6 for a complete list). The templates are all hosted on **Deimos** for storage purposes.

To create a template follow these directions:

1. Install and customize the virtual machine as desired.
   a. Once a template is created it cannot be changed, only remade.
2. Once the virtual machine is complete, right click on the selected virtual machine.
3. Select **Template → Clone to Template**.
   a. It can be converted to a template, but it was preferred to leave the original virtual machine intact and available if changes needed to be made.
4. Name the template to indicate what the template is, and designate the folder to store the template in.
   a. Templates for Dextar are stored in the **Templates** folder.
5. Select the Host and Cluster for where to store the Template.
   a. Templates for DEXTAR were stored in the Mars Cluster in **Deimos** at **172.16.0.4**.
6. Select the correct Datastore and provisioning.
   a. The only datastore on Deimos is **Deimos.**
   b. The recommended, and used, provisioning is **Thin Provisioning**.
      i. This reduces physical hard drive space utilized by the virtual machines.

   c. Please note the **Validation Compatibility** text box. Ensure this reads as **Compatible.**

7. Click **OK** to continue to finishing the template. If all settings are correct, click to finish the template.
8. Depending on current system resource allocations, size of the virtual machine, and the current state of the virtual machine it can take between 10 and 45 minutes to create the template.

Utilize templates in conditions where a lot of virtual machines based off of one OS to reduce the amount of creation time. It is possible to utilize command line scripting to create new machines en masse. However, this is beyond the scope of this manual as DEXTAR did not utilize this methodology.

### Deploying a Virtual Machine from a Template

DEXTAR configuration utilized a series of templates and en masse virtual machine creation was possible due to “spawning” or “spinning out” new virtual machines. This was done from the virtual machine templates created above. To create a new virtual machine from a template:

1. Select the template you wish to create a virtual machine from.
2. Click on the **Getting Started** tab in the right panel.
3. Click on **Deploy to a new virtual machine**.
4. A new window will open similar to if you were installing  new virtual machine.
5. Indicate the name and the location of the new virtual machine.
6. Select **Thin Provisioning** from the storage options.
7. Install the virtual machine.
8. Power the virtual machine on and check the installation.

## Virtual Machine Snapshots

A useful and heavily relied upon tool within the VMware environment is the **Snapshot**. A snapshot is a save point for a virtual machine where the state of the VM is saved and stored within the environment. If problems arise within a VM and there are snapshots in existence, then it would be easy to revert to a previous snapshot instead of reinstalling the VM. The downside is that any information added to the VM after the snapshot was taken will be lost. DEXTAR relies on this function by maintaining two snapshots of each VM within the test scenario. The first is **Before-VM**. This snapshot is the VM before the installation of VMware tools. The second is **Before-Scenario.** This is the snapshot of the machine with VMware tools added and before a scenario is to be run. Because the scenarios within DEXTAR are inherently destructive, the snapshots allow us a simplistic method for reverting the VMs back to their **Before-Scenario** state to save time and increase the functional time of the system.

In order to take a snapshot, you will want to ensure a few things beforehand:

1. Your VM is properly configured to your desired specifications. There is no way to change the snapshot after it is created. You can only create another snapshot. (You may also delete snapshots.)

2. Your VM is not currently in the middle of an operation.
    a. The snapshot will save the current condition, including if the VM is in the middle of an operation. i.e., installing a new piece of software.
3. Your VM is not under any heavy load.
    a. Although the VMware Environment allows for the quiescence of the memory state of a machine, it is the better practice to not have the VM under any load during snapshot.

### Taking a Snapshot

To take a snapshot:

1. Ensure the above steps have been taken.
2. Select the VM that will receive the snapshot and right click.
3. Select **Snapshot → Take a Snapshot.**
4. You will be asked to name the snapshot. Choose an appropriate descriptive name. You may also enter a description in the **Description Box**.
5. At this point you will have the option to save the state and quiescence the memory. This method allows for the snapshot of the stable state in the event that the VM is currently under load. You may choose these options if you wish. They were not used for DEXTAR, instead the above practices were utilized.
6. Click **OK.**
7. The snapshot may take as little as 2 minutes, or as much as 30 minutes dependent on the virtual machine itself.
8. Your snapshot will be completed.

### Reverting to a Snapshot

To revert to a snapshot:

1. If you need to revert to a snapshot, or to manage the snapshots, simply right click on the desired VM and select **Snapshot → Snapshot Manager.**
2. This will open a new window showing you where you are in the snapshot structure, the current snapshot, and any previous snapshots.
3. You may select any snapshot and click **Edit** to edit the name or description.
    a. Please note: you cannot edit the contents of a snapshot. You may only edit the name and description. You may delete and retake a snapshot.
4. Select the desired snapshot to return to.
5. Click **Goto**.
6. This will revert your VM back to the selected snapshot.

## PowerCLI Scripting

This installation contains the VMware PowerCLI (Power Command Line) interface. This interface is a command line implantation that allows for access, modification, and execution of commands in the VMware environment. Every aspect of the environment can be edited and controlled through proper scripting. It is also possible to automate, or run in bulk, a variety of tasks that would take hours if conducted by hand – where it would only take an hour through scripting. This test-bed utilizes only a few scripts, but each are important to the functioning of the test-bed. The PowerCLI is free to download from the VMware website. At the time of writing, the website is:

https://my.vmware.com/web/vmware/details?productId=352&downloadGroup=PCLI550

The scripts utilized in the test bed are listed below in Table 4:

***Table 4. Table of all PowerCLI Scripts and their functions.***

| Script Name | Script Purpose |
|---|---|
| Firewall_Off | Turns all firewalls off within the test network. This is required for a full network scan. |
| Firewall_On | Turns all available firewalls on.  This is the default setting for the test scenario. |
| Power_Off | Powers all VMs within the test scenario off. |
| Power_On | Powers all VMs within the test scenario on. |
| Revert | Reverts all VMs within the test scenario back to the **Before-Scenario** snapshot. |

## Connecting to vCenter

In order for a PowerCLI session to effect the test-bed the session will need to connect to the vCenter Server. It is possible to directly connect to each of the ESXi servers directly, thus bypassing vCenter. However, vCenter is the managing software and should be utilized unless it is no longer possible. To access the vCenter server:

1. Start **PowerCLI.**
2. The initial command prompt displays a few beginning commandlets.
    a. The first is **Connect-VIServer.** This is the one we will use.
3. Type **Connect-VIServer**, after this type the address location of the vCenter server.
    a. For Dextar, **172.16.0.202**
    b. Press **Enter**.
4. After a moment, a windows credentials screen will open. Type in the credentials for the system you are accessing.
    a. DEXTAR vCenter Login: root
    b. Password: D3xt4rT3stb3d!
5. It is common to see server certificate invalidation warnings. These are common and are safely ignored.
6. After login you will be given a command prompt. From here, you can execute command-lets and scripts.
    a. For a full list of commands go to: http://www.vmware.com/support/developer/windowstoolkit/wintk40u1/html/

## Writing a Script

To constrict the length of this manual, scripting will not be covered. There are numerous guides from VMware and private individuals and organizations on the internet to provide a wealth of information regarding this topic. The scripts here were taken from VMware's guide on scripting which can be found on their website within documentation.

The scripts utilized in DEXTAR are in Appendix C.

## Executing a Script

Once a script has been written it can be executed from the PowerCLI command prompt with ease.

1. Navigate to the folder where the desired script is located.
    a. The PowerCLI interface utilizes both Linux and Windows navigation commands.
        i. Windows
            1. cd x = Change Directory to folder X
            2. cd .. = go back one folder
            3. dir = List files and folders in directory
        ii. Linux
            1. cd x = Change Directory to folder X
            2. cd .. = go back one folder
            3. ls = List files and folders in directory
2. To execute type **.\script.ps1** where "script.ps1" is the name of the desired script.
3. The script will execute if programmed correctly.

For most functions, the task window within vSphere will indicate the functions executed by the PowerCLI commands and scripts.

**Please Note:** Scripting allows for the execution of a vast quantity of commands at once. It is entirely possible that such actions can interrupt the operations of the servers, or potentially damage them, even permanently. Please be care that you are aware of what your script is going to do. If your script behaves incorrectly you can stop it by typing **CTRL+C.** This is a universal stop command inside command line interfaces. Alternatively, it may be possible to kill the PowerCLI application and terminate the scripts at the last segment of the script sent. However, it may not be possible to stop an action that the server is actively engaged in and it is NOT advisable to attempt to interrupt a process by powering off a server.

# Test Scenario Construction

In this section, the configuration and construction of the test scenario will be discussed. The premise of the development of the test network was to base it off of current network configuration styles. This was a key element in the design process as the ecological validity of this system is a key element of its strength. There were three main reasons for this decision.

1. The goal is to be ecological valid and to not go against current best practices, or at least common practices if different from best practices. This is to reduce the novelty of the system to experienced participants.
2. The adherence to standard operations of virtual environments reduces training time for participants. This will also increase the performance of participants utilizing environmental structures that they may already be familiar with.
3. The network configuration had to emulate a fairly standard application of a small business network.

This network was developed with a few key aspects in mind that pertain to its size and functionality. First, the network was to emulate a small business of no more than 100 machines. While the DEXTAR system can maintain a larger amount of virtual machines, this first scenario represents a more modest business sized model at 60 machines within the test scenario. In addition to support machines, templates, and backups the virtual machine count is closer to 100. Second, the network had to emulate the complexity of a real physical network. A goal of DEXTAR was to be a self-contained virtual network that could successfully emulate a physical network comprised of various computers, components, and configurations. In order to accomplish this, several different devices and structures were utilized.

The construction of the network happened in stages. Below are the stages, with each respective section discussion each stage:

1. Network Backbone
2. Administrative Access
3. Client Machines
4. Servers
5. Network Scanner
6. Cauldron
7. Threat Generation
8. Intrusion Detection Systems
9. Network Traffic Monitoring
10. Packet Generation
11. PowerCLI

## Network Backbone

In order for the test network to be viable, an internal network structure had to be built. This was accomplished by using a series of switches to control the flow of traffic and add security to a network. Due to the virtual nature, a virtual analogue for a physical switch was required. This was satisfied by the OS called Vyatta. Vyatta is a Linux-based OS that is designed to route traffic, administer firewall rules, and conduct standard router and switch functions. In total, DEXTAR utilizes 9 vyatta switch appliances. They are assigned as illustrated below in Figure 5:

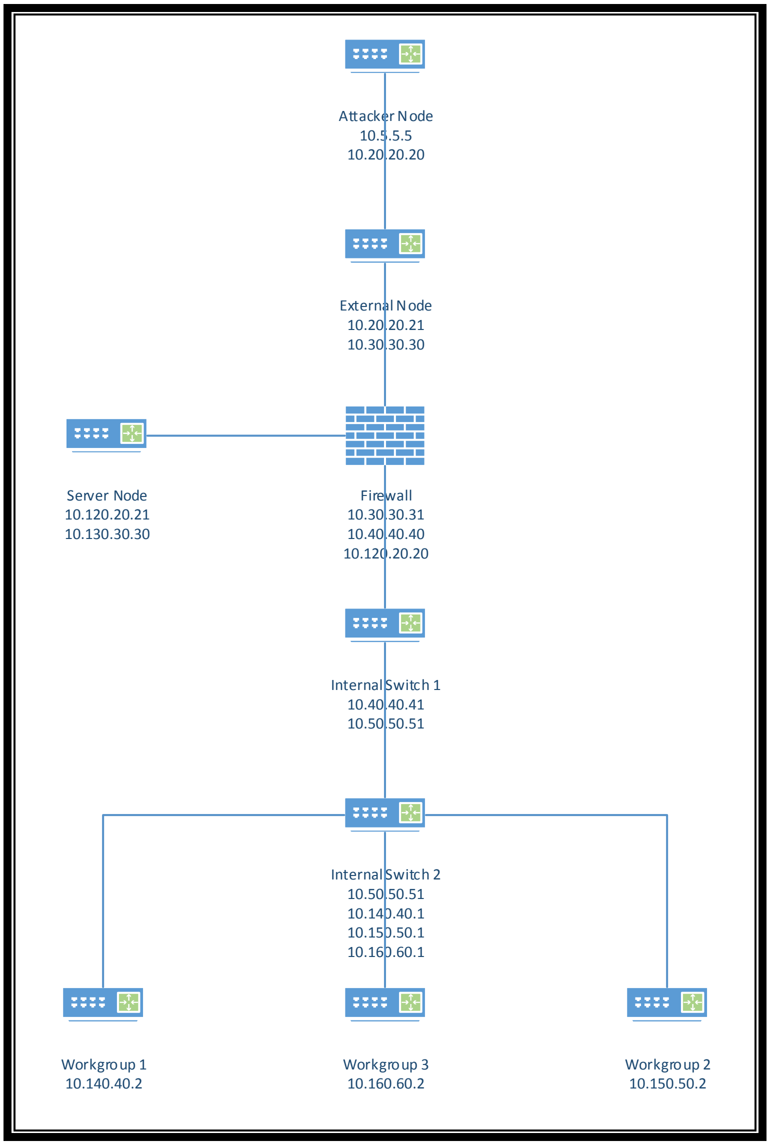


***Figure 5. Vyatta Router Configuration***

The IP addresses assigned to each appliance were arbitrarily assigned after the private network IP range of 10.X.X.X.  This private IP range was required due to the network scanner's limitations, discussed in the section on the network scanner. The sequential pattern of IP address ranges assigned follows a daisy chain pattern for each router to act as the gateway of the previous to eliminate excessive programming. The following table is a list of router IP addresses and description of purpose:

***Table 5. Configuration details of Vyatta Routers.***

| Router Name | IP Addresses | Description |
|---|---|---|

| | | |
|---|---|---|
| Attacker Node | 10.5.5.5; 10.20.20.20 | This node is where experimenters will launch attacks from. It is also consider as "The internet". |
| External Node | 10.20.20.21;<br>10.30.30.30 | This node is to add separation from the attacking node and the firewall entrance of the network. This allows for monitoring. |
| Firewall Node | 10.30.30.31;<br>10.40.40.40;<br>10.120.20.20 | This firewall node acts as the firewall to the network, and also as the firewall between the client workgroups and the servers' node. |
| Server Switch | 10.120.20.21;<br>10.130.30.30 | This switch is for the server group only. |
| Internal Switch 1 | 10.40.40.41;<br>10.50.50.50 | This switch is the first of two internal switches. This separation allows for another layer of monitoring. |
| Internal Switch 2 | 10.50.50.51;<br>10.140.40.1;<br>10.150.50.1;<br>10.160.60.1 | This internal switch is the second of two. This separation allows for monitoring. This switch also diverts traffic to the correct workgroups as needed. |
| Workgroup 1 | 10.140.40.2 | This switch is for the first workgroup only. |
| Workgroup 2 | 10.150.50.2 | This switch is for the second workgroup only. |
| Workgroup 3 | 10.160.60.2 | This switch is for the third workgroup only. |

## Installation of Vyatta

The installation of vyatta is used through a mixture of vSphere processes and setup processes through the vyatta system itself. The software for Vyatta can be found in Deimos Datastore, or downloaded from:

www.vyatta.org/downloads

The required file is the Virtualization ISO. The downloaded version for DEXTAR was VC6.5 – Virtualization ISO. To install follow the installation instructions in the section called "Virtual Machine Installation". There are a few notes for installing vyatta:

1. The installation will require the OS to be marked as: **Other Linux (32-bit)**
2. Vyatta installations will require between 2 and 4 each. This will depend on the router's intended use and design practices.
3. The Vyatta ISO will need to be selected after the VM is created.
    a. The vyatta installation the location is: [Deimos] ISO Storage/vyatta-livecd-virt_VC6.5R1_i386.iso

Once completed, the virtual machine will be constructed, the ISO loaded and ready for installation, and the virtual machine should be on. If those three aspects are not met, revisit these steps to ensure each of the three aspects are met before proceeding.

Each of the 9 vyatta appliances are programmed independently and in a daisy chain fashion. To limit the length of coding, each system acts as the previous system's gateway. Within this, certain rules for traffic flow are installed to ensure the correct path of network traffic. At certain switches the traffic is diverted to the correct location. For example, the Firewall Switch will divert any traffic for 10.130.30.X received by it to that particular subnet and will not let the traffic proceed to another part of the network. In similar fashion, the Internal Switch does this for each of the three workgroups.

For a complete listing of the programming for each machine, please refer to Appendix A where all of the programming is listed. Below is the programming guide for the more complicated Firewall Switch, the basic programming theory is the same among all switches. This covers from the first start up after installation to fully functional.

1. Power on the vyatta installation for the first time.
    a. At this point the Live CD will operate and react as if it is a fully functional system. However it is merely functioning in a solid state in which you cannot permanently change any settings. You will need to install Vyatta from the command line.
2. To log in,  use the following details:
    a. Login: vyatta
    b. Password: vyatta
3. Install vyatta using the following command at the command prompt:

```
install system
```

    a. It should be noted that this is not the indicated command within the Vyatta manuals, however, it is the command that actually installs the system.
    b. During installation you will be able to select a new password. The devices in DEXTAR use the following credentials:
        i. Login: vyatta
        ii. Password: D3xt4rT3stb3d
4. To change a user's password in vyatta, use the following command:

```
set system login user USERNAME authentication plaintext-password
NEWPASSWORD
```

    a. Where USERNAME is the user name whose password you are changing.
    b. Where NEWPASSWORD is the newer password.
5. In order to configure vyatta, you must first go into configuration mode by typing:

```
configure
```

    a. From here, a separate set of commands is available that will allow you to set up the vyatta appliance.
6. To configure the Firewall Switch, we start by assigning the three Ethernet devices addresses.
    a. The first Ethernet device:

```
set interfaces ethernet eth1 address 10.30.30.31/24
```

        i. To explain:

1. Set = instructing vyatta that this is a configuration command to change a particular parameter.
2. Interfaces = instructs vyatta what area to configure
3. Ethernet = instructs which interface
4. Eth1 = Ethernet device, this changes dependent on the interface used (eth1, eth2, etc)
5. Address = specifies what about the eth1 interface is changing
6. X.X.X.X/X = IP address with netmask designation
   a. /24 = equals a netmaks of 255.255.255.0
   b. For more netmasks, use [www.google.com](http://www.google.com)

ii. Once the IP address is assigned, the remaining commands need to be given in individual lines (although presented here together):

```
set interfaces ethernet eth1 duplex auto
set interfaces ethernet eth1 speed auto
set interfaces ethernet eth1 smp-affinity auto
```

1. Each of these commands specify a particular setting within the vyatta appliance.

b. The second Ethernet device:

```
set interfaces ethernet eth2 address 10.40.40.40/24
set interfaces ethernet eth2 duplex auto
set interfaces ethernet eth2 speed auto
set interfaces ethernet eth2 smp-affinity auto
```

c. The third Ethernet device:

```
set interfaces ethernet eth3 address 10.120.20.20/24
set interfaces ethernet eth3 duplex auto
set interfaces ethernet eth3 speed auto
set interfaces ethernet eth3 smp-affinity auto
```

d. At this point, all three Ethernet devices have been assigned. The vyatta appliance will only assign an Ethernet device if there is one installed within the virtual hardware. If additional Ethernet interfaces need to be installed, first install them through the **Edit Settings** menu in vSphere, and then configure them within Vyatta.

e. Once these configuration commands are completed you will need to commit the changes so that the system recognizes and utilizes the new commands. Do this by:

```
commit
```

f. After the system responds that changes have been committed, you will need to save the changes if you wish to keep them. Do this by typing:

```
save
```

    g. The changes above will be committed and saved. You will need to do this after every configuration change.
    h. To leave configuration mode, type

```
exit
```

        i. This will return you to the main command interface. You will not be able to configure Vyatta from this screen.

7. Secondly, we will set the system gateway address:

```
set system gateway-address 10.30.30.30
```

    a. This configuration sets the previous switch as the gateway. This daisy chain method reduces the amount of route table information that must be supplied to each switch.
    b. **Commit** and **Save**.

8. Next, we will configure the routing tables. These tables provide the switch with the appropriate information to route the traffic within the test-bed.
    a. As with the previous configuration, type the **configure** command if you've left configuration mode.
    b. Each line below represents one line in command. This entire table can be copied into the Vyatta appliance through vSphere or individual lines may be written.

```
set protocols static route 10.130.30.0/24 next-hop 10.120.20.21
set protocols static route 10.140.40.0/24 next-hop 10.40.40.41
set protocols static route 10.150.50.0/24 next-hop 10.40.40.41
set protocols static route 10.160.60.0/24 next-hop 10.40.40.41
```

        i. This configuration is piping all information to the server bed through to the server switch at 10.120.20.21. From here, the server switch will direct traffic accordingly.
        ii. This configuration also pipes all workgroup information to the workgroup dummy switch at 10.40.40.41. From here, the workgroup switch will direct traffic accordingly.
    c. **Commit** and **Save.**
    d. **Exit** the configuration session.

9. At this point, the routing tables, and Ethernet devices are set up within Firewall Switch. The programming for the other routers is similar, but with various separate configurations. See Appendix A for a full listing of configurations.
10. If you have incorrectly programmed any particular portion of Vyatta, you can delete the specific configuration in question. To do this replace **set** with **delete**. Once you have entered the command you wish to delete, simply **commit** and **save.**

## Administrative Access

Within DEXTAR there are two main administrative methods. The first and foremost is through the vSphere client on the Gateway machine (Ceri1: 172.16.0.1). The second is through the **Admin VM** stored within **Phobos** running the vSphere client. In both instances the vSphere client is the preferred method of accessing the virtual environment. To download the vSphere client:

1. Open a web browser on a computer connected to the DEXTAR physical network (172.16.0.x).
2. Direct the browser to either Phobos (**http://172.16.0.3**), Deimos (**http://172.16.0.4**), or Eris (**http://172.16.0.202)**.
3. The first website to open will likely show you a certificate error. This is normal as the certificate is generated by the servers and not a "trusted authority". Click **Proceed**.
4. A website will appear with several options. Click on **Download vSphere Client**.
5. Once the download is complete, install the program using the on-screen instructions.

If the vSphere client is unavailable on a networked machine, it is possible to access the vSphere Web Client run by the vCenter Server. To do this:

1. Open a web browser on a computer connected to the DEXTAR physical network (172.16.0.X).
2. Direct the browser to the vCenter Server at **http://172.16.0.202.**
3. The first website to open will likely show a certificate error. This is normal as the certificate is generated by the servers and not a "trusted authority". Click **Proceed**.
4. The website will have a link on the right hand side to the **vSphere Web Client**.
5. Click on **vSphere Web Client.**
6. The same credentials used to log in with the vSphere Desktop Client are the same credentials to log in through the Web Client.

Since the web client is not the preferred access method for DEXTAR, and is prone to slower speeds, this manual will not cover usage of the vSphere Web Client. However, the web client and desktop client have nearly the same functions and structure. It is possible to use either.

The majority of administration will be conducted at the gateway machine (CERI1; 172.16.0.1), however, all functions are capable of being administered within the virtual environment from the Admin VM. This Admin VM houses a copy of vSphere and is accessible through remote desktop connections from inside the test-bed. To access from a linux machine within the test-bed:

1. Open the Remmina Remote Desktop Connection program.
2. Use the login IP: 172.16.0.14
3. Use Login: Dextar_Admin
4. Password: Qnt02Pyu**

These steps allow for the remote access of the Admin VM. This is useful during simulations in which experimenters will want to remain in the room with participants but are in need of adjusting environment conditions or accessing virtual machines.

## Client Workgroups

Following the construction of the network backbone for the test scenario, the client machines were installed. These machines will act as the standard client machine in any network installation. These machines represent employees or stations that are typical users with no special permissions required.

Each workgroup is made up of a variety of Windows XP SP0, Windows XP SP2 + Office, and Ubuntu 12.04 virtual machines. The decision to use Windows XP SP0 was to introduce a variety of possible exploitations into the virtual environment without the need for a user to interact to cause the exploitation to occur. The Windows XP SP2 is used to close those exploitations available in a SP0 installation. However, the addition of MS Office does reintroduce a few other exploitations. The use of Ubuntu was not for exploitation but for variety within the system.

The workgroups are arranged into three workgroups arbitrarily assigned. Workgroup 1 contains 15 machines, workgroup 2 contains 16 machines, and workgroup 3 contains 14 machines. These workgroups are illustrated below in Figure 6, while building on our network map.

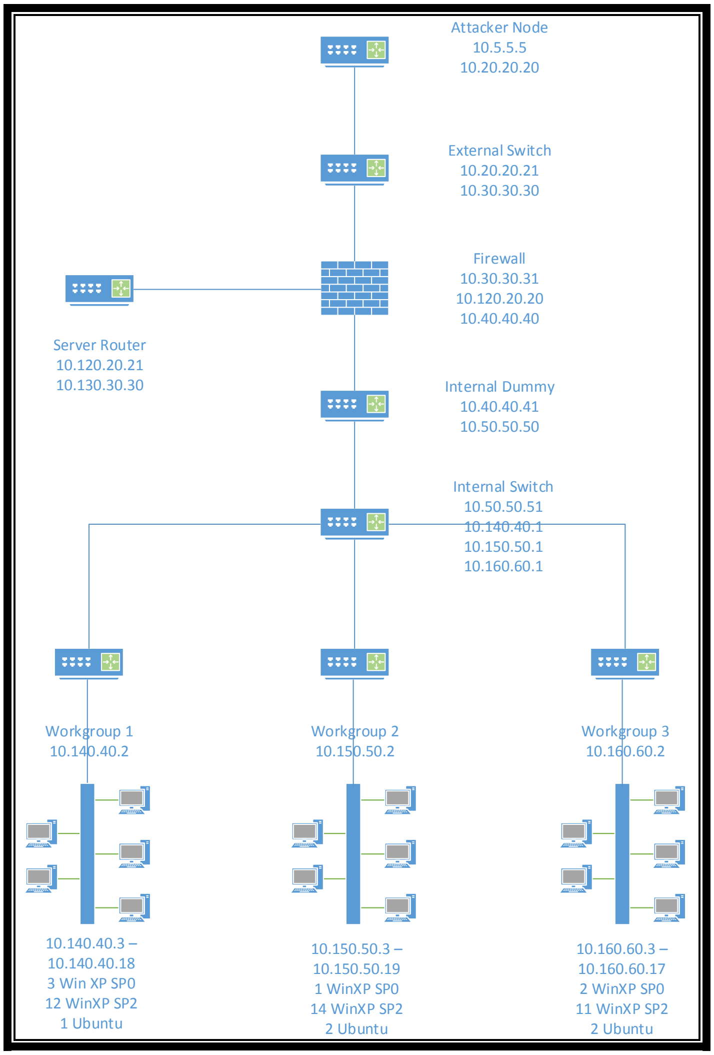


***Figure 6. Workgroup Configuration in DEXTAR Test Scenario***

## Installation of Client Virtual Machines

The installation of the client workgroup machines follows closely to the installation of any other virtual machine. Please follow the instructions in the section titled, “Virtual Machine Installation”. For the customized general hardware settings, the following settings were used:

1. 1 CPU Socket
2. 1-2 Cores
3. 512 MB RAM
4. 20GB – 40GB Thin Provisioned Hard Drive

It should be noted that some machines were installed with varying hardware configurations for variety. The selection of these machines was arbitrary and was only utilized to limit the similarities between all virtual machines. For some installations, it was also necessary to surpass the 20GB hard drive space. Therefore, the larger 40GB space was utilized. However, using Thin Provisioning, this did not allocate a complete 40GB (or 20GB) space to that virtual machine. Instead, the machine uses what it needs and expands as necessary.

The individual virtual machines were built from a series of VM templates. These templates are configured base installations designed for the test scenario. However, DEXTAR operates on more machines than just the test scenario; as such there are a variety of templates available. Those are found below in Table 6:

***Table 6. List of Virtual Machine Templates and Descriptions.***

| Templates | Description |
|---|---|
| BackTrack 5 R3 Base | Base installation of BT for the experimenter use |
| Damn Vulnerable Linux Base | An entirely vulnerable installation of Linux used for penetration testing. Not currently used within DEXTAR. |
| Win 7 IDS | The Windows 7 Intrusion Detection System installation. |
| Win Server 2008 R2 | A base installation with no roles added of Windows Server 2008 Revision 2. |
| Win 7 | A base installation of Windows 7, not activated with license. |
| WinXP SP2 Office 2007 | Base installation of Windows XP Service Pack 2 with Windows Firewall and MS Office 2007 |
| WinXP | A base installation with no firewall or security updates. |

Each machine within the workgroups were named to differentiate them in methods other than IP address labelling. This also adds a layer of validity to modern computer networks in which users often name their computers, or have them named for them. For a complete listing of machines, their templates, and IP address information please see Appendix B.

## Server Workgroup

Within the test scenario, a server workgroup was created as the head of the experimental network. This group contains two servers that are common in smaller organizations. The organization that is emulated in this network is entitled Acme and does not rely heavily on complex network infrastructure. Therefore, they only use two servers to take care of their needs. The first is an Active Directory and DNS server. This server is responsible for logging in both the user, and cataloguing the active computers within the computer directories. The second is a storage server responsible for the storage of the fictional company's network hard drives and their data. The servers added into the test-bed follow the network diagram below in Figure 7:

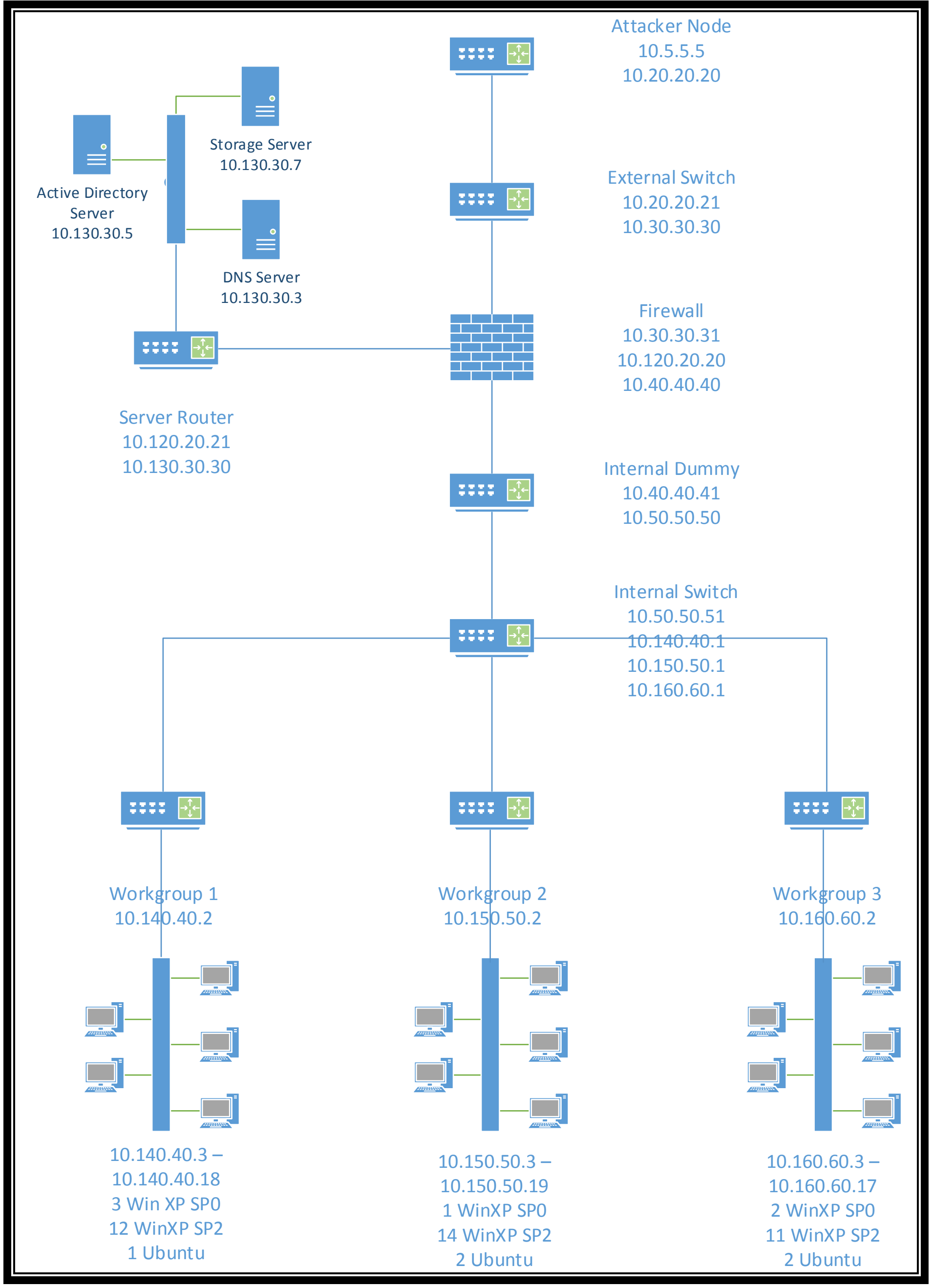


***Figure 7. Network Diagram with Server Group***

The servers were installed using the guide in the section entitled Virtual Machine Installation. Standard installations were chosen with the following hardware specifications.

1. 2 CPUs in 1 Socket
2. 4GB RAM
3. 3 Network Interface Cards
4. 40GB Thin Provisioned Hard Drive

The Active Directory server role was installed using the Windows Server 2008 R2 role installation wizard. The DNS server is an extension of this server, but maintains its own separate address. The following roles were installed:

1. DHCP Server
2. DNS Server (IP: 10.130.30.5)
3. Active Directory Domain Services

In addition to the roles, selected features were added using the feature add wizard:

1. Group Policy Management
2. Remote Server Administration Tools
3. .NET Framework 3.5.1 Features

For each role and feature the standard installation was provided. No special additional rules were applied or customized.

The addition of user accounts to the Active Directory Server was undertaken to allow for a level of validity within the network. The following user accounts are present (Table 7):

***Table 7. User Name, Passwords, and Permission Levels within the Test Scenario***

| User Account | Password | Permissions Level |
|---|---|---|
| John | Password | Standard User |
| Melody | DogGone | Standard User |
| Dawn | DillyDally | Standard User |
| Charlie | Password1 | Power User |
| Amelia | reallylongpassword | Power User |
| Chloe | 123456789 | Standard User |
| Lewis | Iloveyou | Standard User |
| Julia | Sunshine | Power User |
| Logan | Whatwhat247 | Power User |
| Zara | ninjamustard | Power User |
| Aiden | pAsSwOrD | Standard User |
| Rhys | Jenkies27! | Standard User |
| Jake | wonton | Standard User |
| Katie | master | Standard User |
| Molly | 123123 | Standard User |
| Ryan | qwertyitup | Standard User |
| Jakob | JohnJakob | Standard User |
| Jonah | fluffyBunny | Administrator |
| Jax | jesus0000 | Standard User |
| Thomas | Scar | Standard User |
| Dan | Ashbeck | Standard User |

| | | |
|---|---|---|
| Jeremy | Evian27 | Standard User |
| Josh | LochNessie | Power User |
| Layla | Ginger2000 | Standard User |
| Lauren | BuddyRich1 | Standard User |
| Lori | poiuytrewqasdfghjkl | Standard User |
| Isabelle | 753951852456 | Standard User |
| Nicole | 8675309 | Standard User |
| Dexter | password1 | Power User |
| Kaden | steaktartar | Standard User |
| Caleb | jujubeans | Standard User |
| David | Beetlejuice1986 | Standard User |
| Sara | sarap | Standard User |
| Rose | amazon | Standard User |
| Aisha | stupidpassword2013 | Administrator |
| Ellie | qwertyuiopASDFGHJKLzxcvbnm | Administrator |
| Taylor | 1029384756qpwoeiruty | Standard User |
| Stan | shiningStan | Standard User |
| Toby | spidermanrulz | Standard User |
| Daisy | dukesandboots | Power User |
| Johnny | Neopetz4life! | Standard User |
| Abigail | BriarwoodAmber | Standard User |
| Mary | St490dn2k | Standard User |
| Maayan | maneshmachev | Standard User |
| Visitor1 | VisitorAcme | Standard User |

## Network Scanner

Within DEXTAR two network scanners were implemented. The first was the NMAP Scanner freely available at:

http://nmap.org

This is an open source scanner that is heavily utilized within the security community. This scanner can range from a quick superficial scan to a slow in-depth and aggressive scan. For DEXTAR, the middle selection was made. This scanner comes preinstalled within the BackTrack 5 OS, but can be added to any OS platform. The second scanner is the Nessus Vulnerability scanner available for purchase from:

http://www.tenable.com

This scanner was installed into the Admin VM. This scanner is used to assess network vulnerabilities within specific parameters. Although customizable, this scanner is run through a website interface compared to NMAPs command line structure.

The primary scanner used within the network was the NMAP scanner due to the cross-platform structure. Scan data was saved into two formats (*.nmap; *.xml) and provided to Cauldron (See section: Cauldron). The secondary scanner, Nessus, was utilized when possible. However, NMAP provided more detailed scans and was therefore used.

To scan DEXTAR a basic technique was implemented. To utilize more advanced techniques, please refer to the NMAP manual (http://nmap.org/book/man.html) for more information. The following method was used from the BackTrack Red VM:

1. Open **Terminal.**

    a. In BT5, this is an icon on the top menu bar.

2. NMAP is called using the command **nmap** followed by arguments.
    a. A basic scan will be: **nmap X.X.X.X**, where the IP is of the target machine.
    b. A basic range scan will be: **nmap X.X.X.X – Y.Y.Y.Y**, where any range can be specified.
3. The specific commands to analyse the DEXTAR test-bed are:

```
nmap -vvv -O -sTU -oN myscan.nmap -oX myscan.xml --script=vuln --
script targets-traceroute --script-args newtargets --traceroute
target X.X.X.X - X.X.X.Y
```

    a. Workgroups were scanned individually in order to keep NMAP scans incremental.
        i. It is possible to scan multiple ranges by simply adding more ranges to the end of the command.
    b. To explain the above commands:
        i. –vvv = Very Very Verbose mode; this generates a lot of text to provide more information about the scanned system.
        ii. –O = identify the Operating System
        iii. –sTU = complete a TCP and UDP scan
        iv. –oN = Output in .nmap format. A filename must be specified after this argument.
        v. –oX = Output in .xml format. A filename must be specified after this argument.
        vi. --script=vuln = This calls the script family under vulnerabilities. This is a set of 33 different scripts to scan each IP address for vulnerabilities.
        vii. --script targets-traceroute --script-args newtargets --traceroute target = Trace route for topology building
        viii. X.X.X.X – X.X.X.Y = this is the range provided for each workgroup. For example with workgroup1: 10.140.40.1-10.140.40.18.

4. Each scan produced an output file. These files can be read by the Zenmap program, an accompanying program to NMAP that allows for the graphic interface of the scans.

To view the *.nmap scans through Zenmap.

1. The BT5 Red VM and BT5 Admin VM contain the necessary tools to read the scans.
2. Click on **Applications → BackTrack → Information Gathering → Network Analysis → Network Scanner → Zenmap**.
3. A new application window will open.
4. Click on **Scan → Open Scan**. You will need to locate and select your recently completed or previously completed scan(s).
    a. Please Note: Zenmap utilizes the *.XML file format. Cauldron utilizes the *.NMAP format.

5. From here you will be able to explore each found host's information through five different tab options:
   a. Nmap Output – this is the output that Nmap generates during scans.
   b. Ports/Hosts – A list of open ports and the services within them.
   c. Topology – A graphical representation of the scanned targets. This is also colour coded to generally mean:
      i. Green = Secure
      ii. Yellow = Somewhat Secure
      iii. Red = Not Secure
   d. Host Details – This provides information about the host when gained from the network scan.
   e. Scans – The scan arguments used.

It is entirely possible to run a scan through Zenmap. However, during the testing of Zenmap a few argument commands were misinterpreted by Zenamp and the scan was incorrectly run. Therefore, scans were taken directly with the terminal nmap command and fed into Zenmap.

## Cauldron

The DEXTAR test-bed is capable of being a test ground for new software packages meant to aid the cyber domain. One of these tools is called Cauldron. This tool retrieves and analyses scan data from network scanners, such as Nmap and Nessus, and represents these scans in possible attack vectors for an entire network, or division of a network. Cauldron does contain more advanced options, however, only the attack vectors were utilized.

Due to ITAR restrictions, this software was installed on the Admin VM only. All instructions for installation and utilization are available from CyVision (http://cyvisiontechnologies.com).

## Threat Generation

In order to provide realistic threats within the network, a penetration testing OS package developed by Offensive Security and called BackTrack 5 R3 was used. This can be downloaded from:

http://www.backtrack-linux.org/

This OS is freely available and uses only general public licensing. Due to the complexity of the OS and the tools available within it, it is not possible to provide an exhaustive guide into how all threats are generated. Instead, there are training courses offered by Offensive Security into penetration testing and ethical penetration testing that instruct users on how to utilize the tools within BackTrack. Later in this section there will be a guide on how a sample threat is generated.

Within the DEXTAR test-bed only a few type of exploitations were utilized. First, the Windows XP OS was selected due to its original level of vulnerabilities available without user intervention to download a piece of malware. Second, the threats were limited within this testing system to provide stability across experimental conditions. However, this system is fully equipped with all currently available threat definitions publically available from the Metasploit project (http://www.metasploit.com/).

The tool within BT5 utilized primarily is the Armitage GUI for the Metasploit Framework. This interface allowed a point and click navigation method for providing real threats in the test scenario. This tool is capable of providing NMAP scans, Metasploit Meterpreters, Payload delivery, Virus Definitions, and can generate malware laden files. Within DEXTAR we utilize NMAP scans, Metasploit Meterpreters, and Payload deliveries. The specific exploits have been resolved and are permanently disabled in subsequent versions of the Windows operating system.

Below an example threat will be provided, although this is not an exhaustible guide to the generation of all threats – only to the single threat below.

1. Start or log into **BT5 Red VM.**
    a. Login: root
    b. Password: 1qazXSW@3edcVFR$5tgb
2. Go into **Applications → BackTrack → Exploitation Tools → Network Exploitation Tools → Metasploit Framework → Armitage**.
3. When the program starts, Armitage will ask to connect to a server. The default settings are for the localhost and these will function for the purposes of DEXTAR.
4. The second window will ask to start the RPC server for Metasploit. Click **Yes** to proceed.
5. It will take several minutes for all the services to start and Armitage to open.
6. Once open, you can start using the Armitage system. For this demonstration we will infiltrate a single computer using an older explotation: ms08_67_netapi
7. To start, go to **Hosts → Nmap Scans → Quick Scan (OS Detect).**
8. In the new window, we will put in the IP address of a specific machine: **10.160.60.4** (Havoc)
9. The scan reveals a Window XP machine. From here we will go click **OK** to the dialogue box that appears and follow the instructions it provided by selecting the new machine and then go to **Attack → Find Attacks.**
10. After a few moments, another dialogue box will appear, acknowledge this box and continue.
11. **Right click** on the target machine, go into **Attacks → SMB → ms08_67_netapi**
12. A new window will pop up with configuration data. This may be edited to reflect any desired information. For now, click **Launch.**
13. From here, the machine will be captured and Armitage illustrates this by changing the computer icon to red with lightning bolts around it.
14. **Right Click** on the machine again. Go to **Meterpreter 1 → Access → Migrate Now!** This function will migrate the exploit into a new process (notepad.exe) on the target machine.

The example provides a brief demonstration on the GUI interface capabilities of the Armitage interface. Although the metasploit framework is far more powerful than the Armitage interface provides for, a full demonstration of metasploit and its functions can be found on the project website: http://www.metasploit.com.

## Intrusion Detection System

The DEXTAR test-bed utilizes a series of Intrusion Detection Systems and tools to capture the network traffic data for analysis. This analysis aims to identify the ground truth within a given scenario in order to base performance measures from the participants on. The IDS systems will be discussed in this section, while the network traffic monitors will be discussed in the proceeding section.

The IDS systems within DEXTAR are WinIDS based systems. This system was chosen in order to interact with the reporting software and metric system designed for DEXTAR. This reporting system is discussed in the DEXTAR Metrics portion of this manual. The IDS systems lay within the network in one participant accessible position through the Firewall router. This positioning allows the traffic from the "Internet" (for this scenario it is anything outside of the External Switch) to be analysed by the IDS.

The WinIDS system was installed within a standard clone of the Windows Server 2008 virtual machine. The IDS installation was a standard installation of the WinIDS system with an accompanying Wireshark installation (discussed more within the next section). Due to the length of the installation manual it is not reproduced within this manual. The guide may be found at: http://winsnort.com.

The WinIDS comes as a package which includes the SNORT IDS, WinPCAP, Barnyard2 and MySQL. The SNORT IDS is the underlying intrusion detection system that analyses the network traffic captured by WinPCAP program to detect any suspicious activity happening in the network. The SNORT IDS system uses a large set of customizable intrusion rules to flag a set of network activities to be considered a suspicious event. Barnyard2 assists Snort IDS to parse through network traffic data in an effort to capture and analyse 100% of network traffic. Barnyard2 has been configured to write the events generated by the SNORT IDS to the MySQL database. The reporting software, a system installed to collect metric data (further discussed in the "Reporting Software" section) queries the database to retrieve all the alerts and presents it to the analyst for triage. Below in Figure 8, a representation of the structure of the IDS with the reporting software is presented.

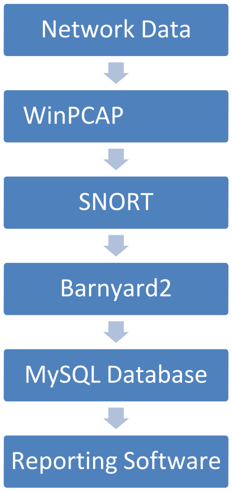


***Figure 8. Intrusion Detection System Structure***

The WinIDS system runs as a windows service within the operating system. This allows the service to be turned on and off. To do this:

1. Click on the **Start Menu**.
2. Go to **Run** and type in **Taskmgr.**
    a. Alternatively, the keystroke **CTRL + Shift + ESC** calls the Task Manager.
3. Navigate to the third tab, **Services.**
4. Located the service **Snort.**

5. **Right Click** on the service.
6. A menu drops down allowing you to turn on or turn off the snort service.

In addition to starting and stopping the SNORT service for the intrusion detection, the database that SNORT writes to will often need to be deleted from a previous run. To do this:

1. Click on the **Start Menu**.
2. Locate the **MySQL Interface**.
    a. This command will initiate a command line terminal.
3. Type the command **use SNORT**.
    a. This command indicates to the interface to reference the SNORT database.
4. To **Save**:
    a. **SNORT > database_file_name.sql**
5. To **Delete**:
    a. **DELETE FROM events where sid > 0;**
    b. **DELETE FROM acid_events where sid>0**
6. Close terminal interface.

The saving and purging of the IDS database is required after each experimental run of the DEXTAR system and is currently not automated through VMware PowerCLI. It is important to ensure that files are written before database deletions occur.

## Network Traffic Monitoring

One of the key components of the DEXTAR test-bed is the ability to capture all of the network traffic that is generated from any source. This is accomplished by a redundant series of Wireshark Network Packet Capture programs placed around the entire network. Wireshark is located in the following locations:

1. Participant Machines
    a. This captures all of the network packets generated by the participants directed at the network.
2. Intrusion Detection System
    a. This captures all of the network packets that the IDS machine receives, which may not include intra-network packet traffic (dependent on conditions).
3. Attacker Machines
    a. This captures all of the network packets generated by the experimenters and directed at the network.
4. Workgroups
    a. There is a separate hidden system within each workgroup designed to capture all network traffic within workgroup, to the workgroup, or from the workgroup.
5. Packet Generators
    a. This captures all of the network packets generated by the packet generator.

The location of these installations is meant to ensure complete network traffic capture. The ability to capture and detect the ground truth within DEXTAR is one of the key advantages of this system for researchers.

Wireshark was installed on each of the physical participant stations by using the technique described in the section: Wireshark, located in the Software for Client Computers section. The operation of this software is straight forward and the example below illustrates a packet capture from a physical participant machine:

1. Ensure that the participant machine is turned on.
    a. Each participant machine is programmed to start Wireshark within the start-up sequence. However, if Wireshark has not started click on the **Shark-Fin Icon** on the left-hand bar to start Wireshark.
2. Once in Wireshark, the main screen provides several options.
    a. Capture
        i. This is the main menu for the capturing of network packet.
    b. Capture Help
        i. This section is dedicated to providing help to use Wireshark.
    c. Files
        i. Allows you to open previous capture files and view sample capture files.
    d. Online
        i. This allows you to visit the project page, display the user's guide, or work with security.
3. For DEXTAR utilization, you will need to **select the capture interface**. For participant machines, this will be listed as **eth0**.
    a. Select the **Eth0** device.
    b. Click **Start.** (Some installations may say "Capture")
4. Once a capture is started, the functions are automatic. It is possible to apply filters to the displayed captured packets without altering the total amount of captured packets.
    a. For instance, it is possible to use the filter **ip.addr==X.X.X.X** to display packets going to or are from the specified IP address.
    b. You may combine filters to provide a more direct filter: **ip.addr==X.X.X.X && ip.addr==Y.Y.Y.Y**.
        i. This filter will display packets containing both IP addresses.
        ii. Conversely, you may replace **&&** with **||** to represent an "or" statement.
5. When packet capture is complete, hit the **Stop icon**.
6. Go to **File → Save As** and save the file according to the study parameters.
7. Close Wireshark.

As the need increases, more tools to capture network traffic may be utilized within the DEXTAR framework from either the virtualization environment APIs or through direct installation into virtual machines. Other notable capture programs are:

1. Cain and Abel
2. Microsoft Network Monitor

3. Tcpdump*
4. Snoop

*It should be noted that **tcpdump** is standard within BackTrack 5 and is therefore present within the test-bed. However, at this time it is not currently utilized.

## Packet Generation

With the DEXTAR test-bed being an Air-gapped system it is impossible to passively allow network traffic to reach levels of ecological validity. Therefore, the DEXTAR test-bed has several packet generation devices to increase the network traffic to provide a level of din within the scenarios. This noise is required for two reasons:

1. The algorithms used within the DEXTAR metrics rely on a level of noise that must be differentiated from the attacks. This allows for a performance measure to be accurately gained.
2. Provides a level of obfuscation within the scenario so that the attacks are not directly presented to the participant thereby nullifying the requirement to search for an attack.

Because of these reasons, the network traffic is randomly generated and distributed to represent a functioning network. At current, the test-bed utilizes two methods of packet generation: Ostinato and Nping.

Ostinato is a GUI based program that allows for multiple streams of packet generation with the capability of assigning the source IP address and destination IP address to each packet. The generator also allows for changing the size of packets, protocols of packets, and layer. Within the DEXTAR scenario, three streams are utilized with one stream to each workgroup. Each stream is spoofed to appear as though it is coming from the servers.

Nping is a command line tool that allows for the generation of packets in some of the same manners as the Ostinato generator. In the DEXTAR test-bed, the Nping generator is used to create packets that create false alerts within the IDS systems to trigger specific events and alert the participant. This is created by using a specific script rather than utilizing a program with control files such as the manner in Ostinato.

### Ostinato

Ostinato was installed onto a clone of Ubuntu 12.04 LTS using the following method:

1. After the clone of an Ubuntu Linux VM the following command was given to download Ostinato:

```
echo ‘deb
http://download.opensuse.org/repositories/home:pstavirs:ostinato/xUb
untu_12.04/ /’ >> /etc/apt/sources.list.d/ostinato.list
```

   a. However, if this doesn’t work right away complete the following.

```
cd /etc/apt/sources.list.d/
sudo gedit ostinato.list
```

   b. Insert the line: “deb http://download.opensuse.org/repositories/home:pstavirs:ostinato/xUbuntu_12.04/ /”
   c. Save, then quit.
   d. Complete the following:

```
wget
http://download.opensuse.org/repositories/home:pstavirs:ostinato/xUb
untu_12.04/Release.key
sudo apt-key add - < Release.key
sudo apt-get update
sudo apt-get install ostinato
```

   e. Proceed with installation.
2. Once installed, start Ostinato:

```
sudo ostinato
```

   a. Please note: Ostinato must be started with root privileges or it will not function correctly. Ostinato will not provide warnings.

In order to send generated packets, use the follow example as a guideline to creating packets.
More instruction can be found on the internet.

1. Ostinato will open with a port group displayed of the localhost (127.0.0.1) and will list enough port groups to display each Ethernet interface and lo.
2. **Select Port 0.**
3. In the top right-hand side of the Ostinato window there will be an open list box just underneath an Apply button and Frames Per Second options. **Right click** inside the list box and select **New Stream.**
4. A new line will be created. **Click on the gear icon** to enter the settings.
5. Under the **Simple** section of Protocol Selection, select the following functions:
   a. Layer 1
      i. MAC
   b. VLAN
      i. Untagged
   c. Layer 2
      i. Ethernet
   d. Layer 3
      i. IPv4
   e. Layer 4
      i. TCP
   f. Layer 5
      i. Text
   g. Payload

        i. Pattern

6. Under the **Protocol Data** tab, use the following settings:
    a. MAC Address
        i. Destination: ff ff ff ff ff ff
            1. This creates a broadcast source allowing the packet to travel the network without specific MAC addresses set for each machine.
        ii. Source: ff ff ff ff ff ff
            1. This creates a broadcast destination allowing the packet to travel the network without a specific MAC address set for each machine.
    b. IPv4
        i. Set the Source and Destination according to the desired targets.
        ii. For this example, set the **source** to **10.130.30.1**
        iii. Set **Source Mode** to **Random Host.**
        iv. Set **Source Count** to **255**.
        v. Set **Source Netmask** to **255.255.255.0**
        vi. Set **Destination** to **10.160.60.3**
        vii. Set **Destination Mode** to **Random Host.**
        viii. Set **Destination Count** to **255**.
        ix. Set **Destination Netmask** to **255.255.255.0**
    c. Payload Data
        i. Set Payload Data to **Random.**
7. Under the **Stream Control** tab use the following settings:
    a. Send
        i. Packets
    b. Numbers
        i. Number of Packets = 25
        ii. This must be greater than 1 or the generator will not select a new host for either the receipt or send.
    c. Rate
        i. Set this to 10 per second.
        ii. This can be adjusted to any rate.
        iii. Please note: the higher per second rating, the more bandwidth and processing power required. It is possible to crash both the virtual machine and network if these settings are too high.
    d. After this Stream
        i. Goto Next Stream
        ii. This section indicates behaviour. If Stop is selected, the generator will stop after the allotted number of packets has been sent. If you will not use multiple streams, use “Goto First” to keep the generator running in a loop.

8. You may view the generated packet style under the **Packet View** tab. However, there are no edits here.
9. Click **OK.**
10. Click **Apply** in the main Ostinato window.
11. Click the **Start Icon** in the lower portion of the Ostinato window to start the stream.
12. Your stream will be transmitting. You may use Wireshark to watch the stream.

### Nping

Nping is installed into the Ostinato VM and is run parallel to Ostinato. This generator is used for the false alerts that are required for proper calculation of performance. This program functions differently than Ostinato and is only available through a terminal interface. To use:

1. Start a **Terminal Window**. This is available on the left-hand bar.
2. The Nping command is a single line. For this example we will use:

```
sudo nping --udp -p 4040 10.160.60.1-40 -c 0 -delay 44400ms
```

   a. This command initiates a packet burst of two packets every 44 seconds to one of the IPs within the given range above using a UDP protocol. The IDS is programmed to flag this type of packet.

3. This stream will continue until manually stopped. To do this use the keystroke **CTRL + C.**

By utilizing both packet generators, the network maintains a level of network traffic through out. During some implementations or scenarios it is possible to use specific data such as transferring a file or a series of files throughout the network. The current scenario does not utilize this method and instead goes with the random generation.

## PowerCLI Scripts

As previously mentioned this system utilizes power scripting to conduct various features and functions to increase the efficiency of the system. These functions are purely maintenance and are outside of the test-scenarios that DEXTAR utilizes at this time. Several scripts are used:

1. Revert to Scenario – base snapshot
   a. This script is responsible for reverting the test-bed to before scenario conditions. This allows for the reconstruction of infiltrated VMs and network states before any damage may have occurred during a previous scenario.
   b. Filename: revert.ps1
2. Power On
   a. Powers on all machines within the test-scenario folder.
   b. Filename: poweron.ps1
3. Power Off
   a. Powers off all machines within the test-scenario folder.
   b. Filename: poweroff.ps1
4. Snapshot

a. Takes a new snapshot of the VMs within the test scenario. This is only used when changes are made to the scenario that need to be persistent.
b. Filename: snapshot.ps1

The specific scripts within each of these scripts are located in the Appendix. For more information on this tool, see the section titled PowerCLI Scripting.

# DEXTAR Metrics

The most powerful aspect of the DEXTAR test-bed is the embedded metrics devised to capture performance, situation awareness, ground truth, and collaboration. This section is dedicated to providing an overview and example of how to access these metrics. A portion of these metrics are currently post-hoc measures and are noted as such. In future upgrades to DEXTAR these measures will become on-line measures and provide real-time performance measures.

There are three categories of metrics used within DEXTAR and three sections within this portion of the manual. They are: 1. Computer-based 2. Audio/Video 3. Collaboration. Each category is important for reasons that may or may not overlap with other categories. For instance, computer based metrics will be able to capture a level of collaboration. However, these captured metrics will possibly be less detailed than specific collaboration data captured by experimenters. It may also be the case that a vast amount of information will be collected through the computer-based metrics that will obfuscate minor, but important, collaborations between participants within a scenario.

## Computer-Based Metrics

The computer-based metrics within DEXTAR rely on the collection and analysis of the network traffic, participant performance, and ground truth within the scenario. We will discuss each of these in some detail as to how they are utilized into useful metrics.

### Network Traffic

Network traffic is captured using the packet capture software deployed around the network. These captures include the entirety of network generated traffic and are made up of several sections:

1. Attacker
2. Intrusion Detection Systems
3. Workgroups
4. Servers

Each of these captures represents a specific area within the network where the capture is collected. Combining each of these sections into a single database allows for the entire network traffic to be viewed. In order to be a viable database, the duplicate alerts (often in multiples as each capture will record the same packet creating up to 7 copies of the same packet) are removed and the database is analysed in comparison. Several analyses are conducted and some examples of these are:

1. Attacker IP filter vs Network Total
   a. This gives us an amount of traffic generated by the Attacker machine in comparison to the remaining network traffic.
   b. In repeated testing this will provide a basis for statistical analysis aiding in the non-parametric methodology utilized within the performance metrics.
   c. It should be noted that the Attacker IP filter is considered in its entirety as the ground truth. More information is added to this collection to expand the ground truth from pivot attacks.
2. IDS Capture vs. Network Total

    a. This provides an efficiency rating of the IDS for a performance review of the IDS.
    b. This also provides for a method of identifying the possible amount of information that participants can utilize from using the IDS system alone. In certain experimental conditions, it may be selected that participants are only allowed to utilized the IDS system and not utilize other network sensors.
3. Workgroups vs IDS
    a. In this arrangement we are highlighting the possibility of traffic to be generated between workgroup computers without crossing the IDS sensor. This is a tactic used within the DEXTAR scenario wherein the attack utilizes local IPs after a pivot table is established allowing for traffic of questionable nature to act only within the subnet provided without crossing the sensor.
    b. This analysis also provides for aiding in ground truth when identifying attack traffic between subnets and within subnets.

There are many possible configurations to analyse the information collected through even just the network traffic, but those highlight a few of the important measures.

## Ground Truth

Ground truth is the base knowledge of what has actually occurred on the network as far as the attack is concerned. This also extends to knowing exactly what the network is doing and sending around. In this system it is possible, and is designed to be so, to capture the ground truth. This is done in a several step process and the creation of multiple databases that are combined later into a single ground truth database. Below are the steps currently taken to ascertain the ground truth:

1. The network packet capture from the Attacker virtual machine adds to the ground truth nearly completely. This capture highlights all of the ‘red movements’ within the network and lead to understanding what has happened.
2. The designed attack is constructed and implemented within a quiet network. This is done by suppressing any IDS alerts that occur from the network that are not associated with the attack. Any suppressible network traffic is also suppressed to get an as clean as possible sample of the network traffic generated solely through the attack.
3. The remaining network traffic is filtered based on the specific attack vector utilized within the scenario in order to gain the clearest capture of the network traffic pertaining solely to the attack. These steps all create the ground truth which is then placed into a single attack vector capture file and provides the analytic tool utilized for ground truth.
4. It should be noted that while each run of the scenario with different participants is treated identically within the experimental conditions, it is not always possible to completely recreate the exact network packets that participants see. Therefore, the creation of a ground truth is required for each run of the scenario. In future implementations of DEXTAR this will be an automated process.

Arguably, the ground truth is one of the most important metrics that the DEXTAR test-bed provides. This forms the basis of comparison of participant performance within the scenario.

This metric alone provides the basis for an enumerable amount of statistical analyses and applications at later stages.

## Participant Performance

Participant performance is based off of the ground truth compiled in the fashion aforementioned. This allows the calculation of performance through such measures as the signal detection theory. This theory has proved in previous systems and applications to be a consistent measurement of performance. There are, however, various other measures of performance within the test-bed. The first and foremost is the reporting system. This system allows for the determination of performance within the scenario, time to completion, correct identification, and situation awareness. The second aspect available to gauge participant performance is based upon the available of additional tools. In the development of the current scenario the visualization tool of Cauldron was introduced and experimentally placed against an open-source counterpart Zenmap. This second aspect of performance is the tool utilization and how it either advanced or hindered the performance of the participant.

### Reporting Software

In DEXTAR, we ask participants to classify IDS alerts as either suspicious activity or not suspicious activity. Whenever a participant defines an alert as suspicious they are provided with a text box that requires them to indicate the reason why it is suspicious. This portion is to help identify the situation awareness a participant has through the following method:

1. The participant is required to categorize alerts that s/he may think are important. This provides a base level of participants acknowledging the alert.
2. The second portion is to require participants to indicate why the alert is suspicious, this provides a level of comprehension.
3. The participant is asked how this alert may lead to a further alert or attribute to a potential attack. This metric provides the basic level of projection.
4. With these three levels, a basic understanding of situation awareness is attained.

The classification of alerts also provides us with the measure of how well the participant has performed, without regard to situation awareness. This is calculated using the signal detection theory and is taken from the overall amount of alerts presented, the number of alerts correctly classified, the number of alerts incorrectly classified, and the number of alerts missed that should have been classified. This makes the basic signal detection equation in which a measure of performance is reached. Further analysis of this information leads to an informed and accurate representation of individual performance, team performance, and study performance.

### Experimental Tools

One of the unique characteristics of this test-bed is the ability to test new, developing, or established tools within the cyber domain for their performance with human participants. The example scenario within this manual uses one such tool called Cauldron (for more information see the section: Cauldron). The only limitation that the DEXTAR system has on the implementation of experimental tools is that of hardware constraints (if a tool requires more resources than available) and licensing constraints (if applicable to the experimental tool requiring the Internet or other verifications not available on an Air-gapped system).

The ability to enhance performance of a cyber-operator in defense is a primary goal of many organizations and governments. Cauldron's goal is to enhance performance by providing potential attack vector graphs to system administrators with the goal of hardening each system and reinforcing connections between systems so that infiltration is improbable. This tool not only provides attack vector graphs, but is based off of network vulnerability scans and therefore can present the vulnerabilities specific machines have.

Although having the ability to test one tool is useful, experimentally there needs to be a control or secondary similar tool to compare the primary tool against. Within the example scenario this secondary tool is Zenmap. An open source tool that allows for the visualization of networks based on network route information, as well as the presentation of the vulnerability data.

Experimentally, these two tools provide a similar function despite Cauldron's advances and attack vector representation. Therefore, the two tools can be placed against one another. The tool utilization and performance difference between the two tools will identify which tool aided in enhanced performance for the cyber-operators.

## Audio/Visual Metrics

The DEXTAR test-bed as currently installed contains a secondary system for recording audio and video. This system is referenced in the beginning of this manual in the section: Audio-Visual System. This system allows for clear recording of audio for each participant on separate channels for later analysis. Overhead, there are cameras aimed at each participant, and a full test-bed view camera, designed to capture the participants as they perform the task. We will discuss these systems separately.

### Audio System

The audio system for DEXTAR is comprised six participant based microphones connected to a central pre-amplifier device. The pre-amplifier then sends the audio data via firewire to the audio recording computer and the DVR to be synced with the video recording (described in the next section). Each of the microphones within this system are noise-cancelling temple mounted microphones. A secondary audio system is in place that allows participants to talk to one another if experimentally needed. This secondary system is not currently implemented within DEXTAR, but is available. The primary system can be represented as seen in Figure 9.

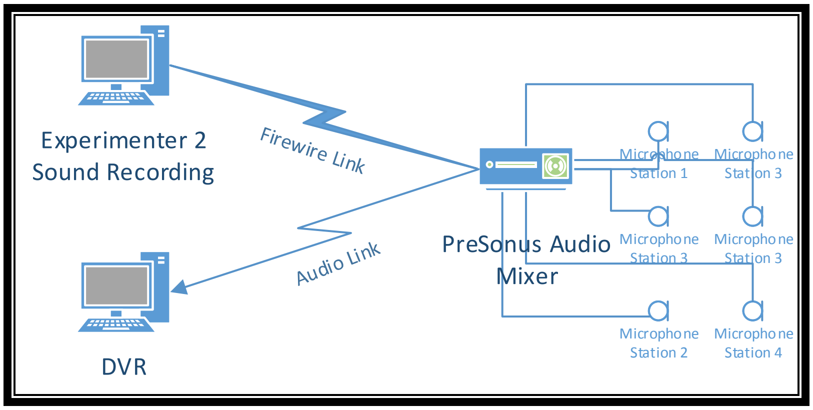


*Figure 9. Audio Recording System*

The audio is recorded using a customized CuBase template designed for this test-bed. Each microphone is recorded to its own channel and can be isolated for specific analysis. The current analytical technique is proprietary software that analyses the audio and indicates the following metrics:

1. Total talk time per participant
2. Total overlap talk time
3. Total recorded time with and without audio presence
4. Total counts on 2,4,6,8, etc. second utterances.
5. Total count of interactions in which two or more participants are speaking to one another.
    a. This metric is also considered to be a part of the collaboration metric. We can identify sections of audio in which participants are interacting and how long the interaction persists.

This allows for the measurement of a team interaction performance score in the presence of team-based scenarios. During single participant performances, this metric is not utilized.

### Video System

The video system implemented in the test-bed is based of six individual oriented overhead cameras and one wide angle overall camera. Each of the six individualized cameras point directly towards a single participant and captures their movements and interactions. The overall camera is a wide-angle view of the entire test-bed area showing the overall interaction of a team of participants. This overall camera is also synced with the audio input to provide an overall recording of the scenario. The camera system can be visualized in this manner as seen in Figure 10.

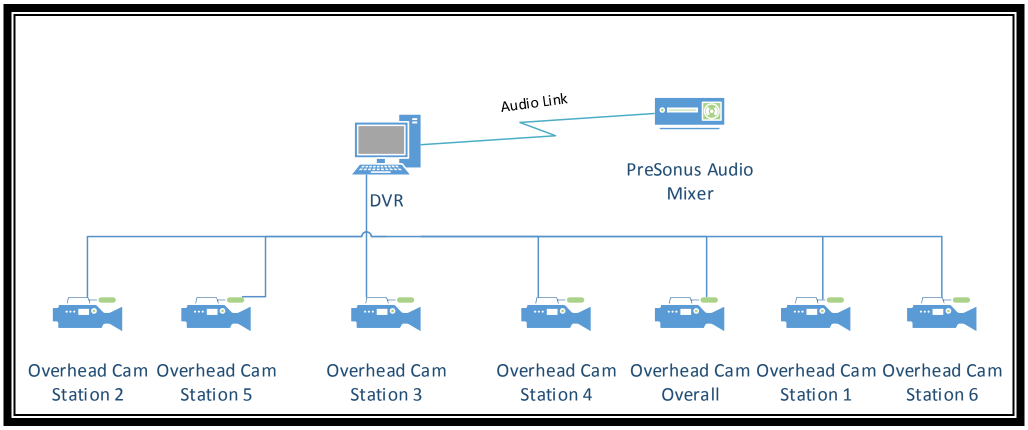


***Figure 10. Video Recording System***

The video recordings allow for a post-hoc confirmation analysis of participant collaboration. This will allow for the validation of interaction and collaboration data collected during the study from the experimenters. It is also possible to use these recordings to review certain portions of a scenario in which a multitude of interactions have occurred that the experimenters will need to parse through.

## Collaboration Metrics

The last key component within the metrics of DEXTAR is the ability to gather collaboration data from participants through computer-based interaction and real-world interaction. The first method, computer-based interaction, is derived through the three main avenues:

1. Presence of classification logs that reference interactions.
2. Information shared between the participants through any electronic means such as Chat, Email, Shared Folder, and so forth.
3. Post experiment questionnaire.

The second method, real-world interaction, is derived from the following sources:

1. Audio recordings
2. Audio analysis
3. Collaboration logger
4. Communication Logger
5. Experimenter Notes outside of Collaboration Logger

We will discuss the Collaboration Logger, Questionnaire, and Information Sharing within this section. The previous section on Audio/Visual Metrics includes a detailed account of how the audio data is collected and represents a method of how collaboration is attained.

### Collaboration Logger

The collaboration logger is a Windows-based program that allows for the categorization and collection of interactions between participants. The information is observed by experimenters and inputted into the system. Information here is collected into a database per participant and per team for later analysis. Post-hoc analysis will indicate the method and pattern of communication.

### Communication Logger

The communication logger is another experimenter-based logging methodology. This logger is used to track the specific communications indicated between participants through categorization. For example, if Participant A indicates, “A port scan occurred on IP 10.130.30.5 from ports 1-150, I’m going to check this out”, the communication logger would indicate that a network event was discussed and would be categorized appropriately. Post-hoc analysis will provide an indication of the most discussed topics.

# Appendix A

*Router Configuration Files*

This appendix contains the configuration settings for each of the 9 vyatta virtual appliances within the DEXTAR test scenario.

## Attacker Router Configuration

```
Interfaces {
        Ethernet eth0 {
                address 10.5.5.5/24
                hw-id 00:50:56:a6:61:e4
        }
        ethernet eth1 {
                address 10.10.10.11/24
                duplex auto
                hw-id
                smp_affinity auto
                speed auto
        }
        ethernet eth2 {
                address 10.20.20.20/24
                duplex auto
                hw-id
                smp_affinity auto
                speed auto
        }
        loopback lo {
        }
}
protocols {
        static {
                route 10.10.10.0/24 {
                        next-hop 10.20.20.21{
                        }
                }
                route 10.140.40.0/24 {
                        next-hop 10.20.20.21{
                        }
                }
                route 10.150.50.0/24 {
                        next-hop 10.20.20.21{
                        }
                }
                route 10.160.60.0/24 {
                        next-hop 10.20.20.21{
                        }
                }
        }
}
system {
        config-management {
                commit-revisions 20
        }
        console {
                device ttyS0 {
                        speed 9600
                }
        }
        gateway-address 10.10.10.10
        host-name vyatta
        login {
                user vyatta {
                authentication {
                        encrypted-password ***************************
                }
                level admin
                }
        }
        ntp {
                server 0.vyatta.pool.ntp.org {
                }
                server 1.vyatta.pool.ntp.org {
                }
                server 2.vyatta.pool.ntp.org {
                }
        }
        package {
                auto-sync 1
                repository community {
                        components main
                        distribution stable
                        password “”
                        url http://packages.vyatta.com/yatta
                        username “”
                }
        }
        syslog {
                global {
                        facility all {
                                level notice
                        }
                        facility protocols {
```

```
                        level debug
                    }
                }
            }
            time-zone GMT
}
```

Attack Router login: vyatta
password: D3xt4rT3stb3d

## External Router Configuration

```
Interfaces {
        ethernet eth1 {
                address 10.20.20.21/24
                duplex auto
                hw-id
                smp_affinity auto
                speed auto
        }
        ethernet eth2 {
                address 10.30.30.30/24
                duplex auto
                hw-id
                smp_affinity auto
                speed auto
        }
        loopback lo {
        }
}
protocols {
        static {
                route 10.100.10.0/24 {
                        next-hop 10.30.30.31{
                        }
                }
                route 10.140.40.0/24 {
                        next-hop 10.30.30.31{
                        }
                }
                route 10.150.50.0/24 {
                        next-hop 10.30.30.31{
                        }
                }
                route 10.160.60.0/24 {
```

```
            next-hop 10.30.30.31{
            }
        }
    }
}
system {
    config-management {
        commit-revisions 20
    }
    console {
        device ttyS0 {
            speed 9600
        }
    }
    gateway-address 10.20.20.20
    host-name vyatta
    login {
        user vyatta {
        authentication {
            encrypted-password ************************
        }
        level admin
        }
    }
    ntp {
        server 0.vyatta.pool.ntp.org {
        }
        server 1.vyatta.pool.ntp.org {
        }
        server 2.vyatta.pool.ntp.org {
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
            password “”
            url http://packages.vyatta.com/yatta
            username “”
        }
    }
    syslog {
        global {
            facility all {
                level notice
            }
            facility protocols {
                level debug
            }
        }
    }
    time-zone GMT
}
```

External Router Login: vyatta
password: D3xt4rT3stb3d

## Firewall Router Configuration

```
Interfaces {
    ethernet eth1 {
        address 10.30.30.31/24
        duplex auto
        hw-id
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 10.40.40.40/24
        duplex auto
        hw-id
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        address 10.120.20.20/24
        duplex auto
        hw-id
        smp_affinity auto
        speed auto
    }

    loopback lo {
    }
}
protocols {
    static {
        route 10.130.30.0/24 {
            next-hop 10.120.20.21{
            }
        }
        route 10.140.40.0/24 {
            next-hop 10.40.40.41{
            }
        }
        route 10.150.50.0/24 {
            next-hop 10.40.40.41{
            }
        }
        route 10.160.60.0/24 {
            next-hop 10.40.40.41{
            }
        }
    }
}
system {
    config-management {
        commit-revisions 20
    }
    console {
        device ttyS0 {
            speed 9600
        }
    }
    gateway-address 10.30.30.30
    host-name vyatta
    login {
        user vyatta {
        authentication {
            encrypted-password *************************
        }
        level admin
        }
    }
    ntp {
        server 0.vyatta.pool.ntp.org {
        }
        server 1.vyatta.pool.ntp.org {
        }
        server 2.vyatta.pool.ntp.org {
        }
    }
    package {
        auto-sync 1
        repository community {
            components main
            distribution stable
```

```
                        password “”
                        url http://packages.vyatta.com/yatta
                        username “”
                }
        }
        syslog {
                global {
                        facility all {
                                level notice
                        }
                        facility protocols {
                                level debug
                        }
                }
        }
        time-zone GMT
}
```

External Router Login: vyatta
password: D3xt4rT3stb3d

## DMZ Router Configuration

```
Interfaces {
        ethernet eth1 {
                address 10.120.20.21/24
                duplex auto
                hw-id
                smp_affinity auto
                speed auto
        }
        ethernet eth2 {
                address 10.130.30.30/24
                duplex auto
                hw-id
                smp_affinity auto
                speed auto
        }
        loopback lo {
        }
}
protocols {
        static {
                route 10.140.40.0/24 {
                        next-hop 10.120.20.20{
                        }
                }
                route 10.150.50.0/24 {
                        next-hop 10.120.20.20{
                        }
                }
                route 10.160.60.0/24 {
                        next-hop 10.120.20.20{
                        }
                }
        }
}
system {
        config-management {
                commit-revisions 20
        }
        console {
                device ttyS0 {
                        speed 9600
                }
        }
        gateway-address 10.120.20.20
        host-name vyatta
        login {
                user vyatta {
                authentication {
                        encrypted-password *************************
                }
                level admin
                }
        }
        ntp {
                server 0.vyatta.pool.ntp.org {
                }
                server 1.vyatta.pool.ntp.org {
                }
                server 2.vyatta.pool.ntp.org {
                }
        }
        package {
                auto-sync 1
                repository community {
                        components main
                        distribution stable
                        password “”
                        url http://packages.vyatta.com/yatta
                        username “”
                }
        }
        syslog {
                global {
                        facility all {
                                level notice
                        }
                        facility protocols {
                                level debug
                        }
                }
        }
        time-zone GMT
}
```

External Router Login: vyatta
password: D3xt4rT3stb3d

## Internal Router Configuration

```
Interfaces {
        ethernet eth1 {
                address 10.40.40.41/24
                duplex auto
                hw-id
                smp_affinity auto
                speed auto
        }
        ethernet eth2 {
                address 10.50.50.50/24
                duplex auto
                hw-id
                smp_affinity auto
                speed auto
        }
        loopback lo {
        }
}
protocols {
        static {
                route 10.130.30.0/24 {
                        next-hop 10.40.40.40{
                        }
                }
                route 10.140.40.0/24 {
                        next-hop 10.50.50.50{
                }
            }
            route 10.150.50.0/24 {
                next-hop 10.50.50.50{
                }
            }
            route 10.160.60.0/24 {
                next-hop 10.50.50.50{
                }
            }
        }
}
system {
        config-management {
            commit-revisions 20
        }
        console {
            device ttyS0 {
                speed 9600
            }
        }
        gateway-address 10.40.40.40
        host-name vyatta
        login {
            user vyatta {
            authentication {
                encrypted-password *************************
            }
            level admin
            }
        }
        ntp {
            server 0.vyatta.pool.ntp.org {
            }
            server 1.vyatta.pool.ntp.org {
            }
            server 2.vyatta.pool.ntp.org {
            }
        }
        package {
            auto-sync 1
            repository community {
                components main
                distribution stable
                password “”
                url http://packages.vyatta.com/yatta
                username “”
            }
        }
        syslog {
            global {
                facility all {
                    level notice
                }
                facility protocols {
                    level debug
                }
            }
        }
        time-zone GMT
}
```

External Router Login: vyatta
password: D3xt4rT3stb3d

## Internal_Switch Router Configuration

```
Interfaces {
    ethernet eth1 {
        address 10.50.50.51/24
        duplex auto
        hw-id
        smp_affinity auto
        speed auto
    }
    ethernet eth2 {
        address 10.140.40.1/24
        duplex auto
        hw-id
        smp_affinity auto
        speed auto
    }
    ethernet eth3 {
        address 10.150.50.1/24
        duplex auto
        hw-id
        smp_affinity auto
        speed auto
    }
    ethernet eth4 {
        address 10.160.60.1/24
        duplex auto
```

```
                hw-id
                smp_affinity auto
                speed auto
        }

        loopback lo {
        }
}
protocols {
        static {
                route 10.130.30.0/24 {
                        next-hop 10.40.40.40{
                        }
                }
                route 10.140.40.0/24 {
                        next-hop 50.50.50.50{
                        }
                }
                route 10.150.50.0/24 {
                        next-hop 10.50.50.50{
                        }
                }
                route 10.160.60.0/24 {
                        next-hop 10.50.50.50{
                        }
                }
        }
}
system {
        config-management {
                commit-revisions 20
        }
        console {
                device ttyS0 {
                        speed 9600
                }
        }
        gateway-address 10.40.40.40
        host-name vyatta
        login {
                user vyatta {
                authentication {
                        encrypted-password ************************
                }
                level admin
                }
        }
        ntp {
                server 0.vyatta.pool.ntp.org {
                }
                server 1.vyatta.pool.ntp.org {
                }
                server 2.vyatta.pool.ntp.org {
                }
        }
        package {
                auto-sync 1
                repository community {
                        components main
                        distribution stable
                        password “”
                        url http://packages.vyatta.com/yatta
                        username “”
                }
        }
        syslog {
                global {
                        facility all {
                                level notice
                        }
                        facility protocols {
                                level debug
                        }
                }
        }
        time-zone GMT
}
```

External Router Login: vyatta
password: D3xt4rT3stb3d

# Appendix B

*Test Scenario Virtual Machine Information Tables*

This appendix contains the individual information for each virtual machine within the test scenario network.

| Virtual Machine Name | IP Address | Template |
|---|---|---|
| Acme Rule | 10.130.30.5 | Windows Server 2008 R2 |
| Acme Storage Server | 10.130.30.7 | Windows Server 2008 R2 |
| IDS | 10.50.50.52 | Windows 7 |
| Sherry1 | 10.140.40.3 | Windows XP |
| Gene | 10.140.40.4 | Windows XP SP2 w/ Office |
| Jerry | 10.140.40.5 | Windows XP SP2 w/ Office |
| Billy | 10.140.40.6 | Windows XP |
| Derek | 10.140.40.7 | Windows XP |
| Bob | 10.140.40.8 | Windows XP SP2 w/ Office |
| Chris | 10.140.40.9 | Windows XP SP2 w/ Office |
| Carrie | 10.140.40.10 | Windows XP SP2 w/ Office |
| David | 10.140.40.11 | Windows XP SP2 w/ Office |
| Geronimo | 10.140.40.12 | Windows XP SP2 w/ Office |
| Francis | 10.140.40.13 | Windows XP SP2 w/ Office |
| Ian | 10.140.40.14 | Windows XP SP2 w/ Office |
| Debbie | 10.140.40.15 | Windows XP SP2 w/ Office |
| Joe | 10.140.40.16 | Windows XP SP2 w/ Office |
| Erin | 10.140.40.17 | Windows XP SP2 w/ Office |
| Jake | 10.140.40.18 | Ubuntu 12.04 |
| Tardis | 10.150.50.3 | Windows XP SP2 w/ Office |
| Dalek | 10.150.50.4 | Windows XP SP2 w/ Office |
| Gallifrey | 10.150.50.5 | Windows XP SP2 w/ Office |
| Davros | 10.150.50.6 | Windows XP SP2 w/ Office |
| Cybermen | 10.150.50.7 | Windows XP |
| Idris | 10.150.50.8 | Windows XP SP2 w/ Office |
| Tish | 10.150.50.9 | Windows XP SP2 w/ Office |
| Castellan | 10.150.50.10 | Windows XP SP2 w/ Office |
| Strax | 10.150.50.11 | Windows XP SP2 w/ Office |
| Trinity | 10.150.50.12 | Windows XP SP2 w/ Office |
| Nerys | 10.150.50.13 | Windows XP SP2 w/ Office |
| Icthar | 10.150.50.14 | Windows XP SP2 w/ Office |
| Rassilon | 10.150.50.15 | Windows XP SP2 w/ Office |
| Donna | 10.150.50.16 | Windows XP SP2 w/ Office |
| River | 10.150.50.17 | Windows XP SP2 w/ Office |
| Ood | 10.150.50.18 | Ubuntu 12.04 |
| Beaux | 10.150.50.19 | Ubuntu 12.04 |
| Renec1 | 10.160.60.3 | Windows XP |
| Havoc1 | 10.160.60.4 | Windows XP |
| George1 | 10.160.60.5 | Windows XP SP2 w/ Office |
| Samson1 | 10.160.60.6 | Windows XP SP2 w/ Office |
| CircumNavigate1 | 10.160.60.7 | Windows XP SP2 w/ Office |
| Phil1 | 10.160.60.8 | Windows XP SP2 w/ Office |
| Tamsin1 | 10.160.60.9 | Windows XP SP2 w/ Office |

| | | |
|---|---|---|
| Vex1 | 10.160.60.10 | Windows XP SP2 w/ Office |
| Dyson1 | 10.160.60.11 | Windows XP SP2 w/ Office |
| Kinsey1 | 10.160.60.12 | Windows XP SP2 w/ Office |
| Hale1 | 10.160.60.13 | Windows XP SP2 w/ Office |
| Astraea1 | 10.160.60.14 | Windows XP SP2 w/ Office |
| Trick1 | 10.160.60.15 | Windows XP SP2 w/ Office |
| Cruz | 10.160.60.16 | Ubuntu 12.04 |
| Bop | 10.160.60.17 | Ubuntu 12.04 |

# Appendix C

*DEXTAR PowerCLI Scripts*

This appendix contains the various scripts utilized within the DEXTAR test-bed to complete various required functions.

## Power On Script

This script powers on each individual VM within the DEXTAR Scenario.

```
#Runs a power on script to start the test scenario one section at a time.

#Attackers
$VMs = Get-Folder 'Attackers' | Get-VM
foreach ( $vm in $VMs ) {Start-VM $vm  -confirm:$false -ErrorAction SilentlyContinue}

#IDS
$VMs = Get-Folder 'IDS' | Get-VM
foreach ( $vm in $VMs ) {Start-VM $vm  -confirm:$false -ErrorAction SilentlyContinue}

#Packet Generators
$VMs = Get-Folder 'Packet Generators' | Get-VM
foreach ( $vm in $VMs ) {Start-VM $vm  -confirm:$false -ErrorAction SilentlyContinue}

#Routers
$VMs = Get-Folder 'Routers' | Get-VM
foreach ( $vm in $VMs ) {Start-VM $vm  -confirm:$false -ErrorAction SilentlyContinue}

#Servers
$VMs = Get-Folder 'Servers' | Get-VM
foreach ( $vm in $VMs ) {Start-VM $vm  -confirm:$false -ErrorAction SilentlyContinue}

#Workgroup 1
$VMs = Get-Folder 'Workgroup 1' | Get-VM
foreach ( $vm in $VMs ) {Start-VM $vm  -confirm:$false -ErrorAction SilentlyContinue}

#Workgroup 2
$VMs = Get-Folder 'Workgroup 2' | Get-VM
foreach ( $vm in $VMs ) {Start-VM $vm  -confirm:$false -ErrorAction SilentlyContinue}

#Workgroup 3
$VMs = Get-Folder 'Workgroup 3' | Get-VM
foreach ( $vm in $VMs ) {Start-VM $vm  -confirm:$false -ErrorAction SilentlyContinue}

#End script.
```

## Power Off Script

This script powers off all VMs within the DEXTAR Scenario.

```
#Run this script to power off all VMs within the Scenario.
$VMs = Get-Folder 'DEXTAR Scenario Test-bed' | Get-VM
foreach ( $vm in $VMs ) {Stop-VM $vm  -confirm:$false -RunAsync}

# End script.
```

## Reversion Script

In order to continually utilize the DEXTAR Scenario for each team while replicating the network for consistency, this script reverts the scenario to a pre-scenario snapshot.

```
# Start script

#Resets all scenario to "Before-Scenario" snapshot.

#Server Group.
$VMs = Get-Folder 'Servers' | Get-VM
foreach ( $vm in $VMs ) {Set-VM -VM $vm -Snapshot 'Before-Scenario' -confirm:$false}
foreach ( $vm in $VMs ) {Start-VM $vm -confirm:$false -ErrorAction SilentlyContinue}

#Workgroup 1.
$VMs = Get-Folder 'Workgroup 1' | Get-VM
foreach ( $vm in $VMs ) {Set-VM -VM $vm -Snapshot 'Before-Scenario' -confirm:$false}
foreach ( $vm in $VMs ) {Start-VM $vm -confirm:$false -ErrorAction SilentlyContinue}

#Workgoup 2.
$VMs = Get-Folder 'Workgroup 2' | Get-VM
foreach ( $vm in $VMs ) {Set-VM -VM $vm -Snapshot 'Before-Scenario' -confirm:$false}
foreach ( $vm in $VMs ) {Start-VM $vm -confirm:$false -ErrorAction SilentlyContinue}

#Workgroup 3.
$VMs = Get-Folder 'Workgroup 3' | Get-VM
foreach ( $vm in $VMs ) {Set-VM -VM $vm -Snapshot 'Before-Scenario' -confirm:$false}
foreach ( $vm in $VMs ) {Start-VM $vm -confirm:$false -ErrorAction SilentlyContinue}

# End Script
```

## Snapshot Script

This script was utilized to take new snapshots of the scenario virtual machines before testing. It is imperative that snapshots not be deleted, but instead new snapshots are taken and stored. This allows for a chain of reversion.

```
#This script is for creating a new snapshot.

#Attackers
$VMs = Get-Folder 'Attackers' | Get-VM
foreach ( $vm in $VMs ) {New-Snapshot -Name Before-Scenario -VM $vm}

#IDS
$VMs = Get-Folder 'IDS' | Get-VM
```

```
foreach ( $vm in $VMs ) {New-Snapshot -Name Before-Scenario -VM $vm}

#Packet Generators
$VMs = Get-Folder 'Packet Generators' | Get-VM
foreach ( $vm in $VMs ) {New-Snapshot -Name Before-Scenario -VM $vm}

#Routers
$VMs = Get-Folder 'Routers' | Get-VM
foreach ( $vm in $VMs ) {New-Snapshot -Name Before-Scenario -VM $vm}

#Servers
$VMs = Get-Folder 'Servers' | Get-VM
foreach ( $vm in $VMs ) {New-Snapshot -Name Before-Scenario -VM $vm}

#Workgroup 1
$VMs = Get-Folder 'Workgroup 1' | Get-VM
foreach ( $vm in $VMs ) {New-Snapshot -Name Before-Scenario -VM $vm}

#Workgroup 2
$VMs = Get-Folder 'Workgroup 2' | Get-VM
foreach ( $vm in $VMs ) {New-Snapshot -Name Before-Scenario -VM $vm}

#Workgroup 3
$VMs = Get-Folder 'Workgroup 3' | Get-VM
foreach ( $vm in $VMs ) {New-Snapshot -Name Before-Scenario -VM $vm}
#End script.
```